UNIVERSIDADE FEDERAL RURAL DO SEMI-ÁRIDO
CAMPUS CARAÚBAS
CURSO DE CIÊNCIA E TECNOLOGIA

JOSÉ GARIBALDI DUARTE JÚNIOR

DESENVOLVIMENTO DE UM DEMODULADOR BFSK EM FPGA

CARAÚBAS – RN
2016

JOSÉ GARIBALDI DUARTE JÚNIOR

DESENVOLVIMENTO DE UM DEMODULADOR BFSK EM FPGA

Monografia apresentada à Universidade Federal Rural do Semi-Árido (UFERSA), como exigência final para obtenção do título de Bacharel em Ciência e Tecnologia.

Orientador: Francisco de Assis Brito Filho – UFERSA.

CARAÚBAS – RN
2016

D812d Duarte Júnior, José Garibaldi.
Desenvolvimento de um demodulador BFSK em FPGA / José Garibaldi Duarte Júnior. - 2016.
59 f. : il.

Orientador: Francisco De Assis Brito Filho.
Monografia (graduação) - Universidade Federal Rural do Semi-árido, Curso de Ciência e Tecnologia, 2016.

1. Modulação. 2. Demodulação digital de frequência. 3. FPGA. 4. Linguagem de descrição de hardware. I. De Assis Brito Filho, Francisco, orient. II. Título.

JOSÉ GARIBALDI DUARTE JÚNIOR

DESENVOLVIMENTO DE UM DEMODULADOR BFSK EM FPGA

Monografia apresentada à Universidade Federal Rural do Semi-Árido (UFERSA), como exigência final para obtenção do título de Bacharel em Ciência e Tecnologia.

Orientador: Francisco de Assis Brito Filho – UFERSA.

APROVADO EM 11 / 05 / 16

BANCA EXAMINADORA

[redacted]

Prof°. Dr. Francisco de Assis Brito Filho - UFERSA

Orientador

[redacted]

Prof°. Dr. Valdemir Praxedes da Silva Neto - UFERSA

[redacted]

Prof° Dr. Hugo Michel Câmara de Azevedo Maia - UFERSA

AGRADECIMENTOS

Agradeço aos meus pais, Garibaldi e Bernadete, por todo o amor incondicional, pelo cuidado, apoio, pelos ensinamentos, e por sempre estarem ao meu lado, não deixando faltar nada.

A minha avó Francisca Matilde, por sempre se mostrar como uma mãe, não poupando cuidado e atenção em todos os momentos.

A minha família como um todo, por sempre se mostrar unida diante das dificuldades e ter possibilitado o meu crescimento, tanto na vida acadêmica como cidadã.

A minha companheira Alice Andrade, por todo amor a mim oferecido, por estar sempre ao meu lado nos bons e maus momentos, pelo carinho, cuidado e compreensão.

Ao meu orientador Professor Brito Filho, por todo o esforço e confiança em mim depositada para a realização deste trabalho, pela perseverança em me conduzir sempre no caminho certo, por toda a orientação e dedicação.

Aos meus professores, tanto na vida escolar como universitária, por estimularem o meu aprendizado ao longo dos anos.

Aos meus amigos, por sempre estarem ao meu lado, por todo o apoio e confiança.

RESUMO

O processamento e transmissão de informação através de sinais modulados é uma dos objetos de estudo mais importantes da telecomunicação, e o estudo de novas técnicas agrega bastante a esse ramo da tecnologia. Este trabalho visa fazer uma análise a cerca do processo de modulação e demodulação de sinais digitais, dando ênfase à recuperação de dados digitais modulados na frequência. É feito, inicialmente, um estudo sobre os sistemas de modulação, analógicos e digitais mais básicos, e em seguida, é proposto a utilização de linguagem de descrição de *hardware* para produzir um demodulador *BFSK* embarcado em *FPGA*. É utilizado um *Phase Locked Loop* digital para implementar o demodulador. Por fim, é verificado o funcionamento do projeto, primeiro, a partir de simulações computacionais, e, posteriormente, através de experimentos utilizando *kit* de desenvolvimento em *FPGA*. Verifica-se, então, que a utilização de sistemas programáveis para aquisição de informação modulada é uma forma eficiente para a demodulação de sinais digitais.

Palavras-Chave: Modulação, Demodulação, *BFSK*, *ADPLL*, *FPGA*.

ABSTRACT

The processing and transmission of information through modulated signals is study object of the most important areas of telecommunications, and the study of new techniques rather adds to this branch of technology. This work aims to make an analysis of the process of modulation and demodulation of digital signals, giving emphasis to the recovery of digital data modulated in frequency. It is made initially, a study on modulation systems, analog and digital, more basic, and then the use of hardware description language is proposed to produce a BFSK demodulator embedded in FPGA. It used a digital Phase Locked Loop to implement the demodulator. Finally, the operation of the project is checked first from computer simulations and later, through experiments using FPGA development kit. It is verified then the use of programmable acquisition systems modulated information is an effective way to demodulation of digital signals.

Keywords: Modulation, Demodulation, BFSK, ADPLL, FPGA.

LISTA DE ABREVIATURAS

AD – Conversão analógica para digital

ADPLL – *All Digital Phase Locked Loop*

AM – *Amplitude Modulation*

ASK – *Amplitude Shift Keying*

BFSK – *Binary Frequency Shift Keying*

CI – Circuito Integrado

CLB - *Configurable Logic Block*

DCO – *Digitally Controlled Oscillator*

DPLL – *Digital Phase Locked Loop*

ECPD – *Edge Controlled Phase Dectector*

FM – *Frequency Modulation*

FPGA – *Field Programmable Gate Array*

FSK – *Frequency Shift Keying*

M-ASK – *Multi-Amplitude Shift Keying*

OOK – *On-off Keying*

PFD – *Phase/Frequency Detector*

PLL – *Phase Locked Loop*

PM – *Phase Modulation*

PSK – *Phase Shift Keying*

QPSK – *Quadrature Phase Shift Keying*

ULA – Unidade Lógica Aritmética

VCO – *Voltage-controlled Oscillator*

VHDL – *VHSIC Hardware Description Language*

LISTA DE SÍMBOLOS

E_o – Amplitude de um sinal variável no tempo

f_1 – Frequência menor de um sinal modulado na frequência

f_2 – Frequência maior de um sinal modulado na frequência

f_c – Frequência central

f_o – Frequência inicial

KM – Constante de modulação

k_o – Constante inicial de modulação

Δf_H – Amplitude de frequência de sinal modulado

Δf – Variação de frequência

K – Parâmetro de multiplicação de frequência do Filtro de *Loop*

M – Parâmetro multiplicativo para um *ADPLL*

$m(t)$ – Informação variante no tempo

N – Parâmetro de multiplicação de frequência do *DCO*

Φ – Fase de um sinal senoidal

$p(t)$ – Portadora senoidal variante no tempo

$x(t)$ – Informação

LISTA DE FIGURAS

SUMÁRIO

1. INTRODUÇÃO

Em sistemas de telecomunicação e processamento de sinais, a modulação em frequência (*FM*) é a codificação de informações sobre uma onda portadora através da variação instantânea da frequência da portadora. As informações podem ser codificadas e transmitidas por meio da onda portadora através da mudança da frequência da portadora entre um conjunto pré-definido de frequências (PATEL, 2013). O oposto do processo descrito é chamado de demodulação. A modulação-demodulação (MODEM) são amplamente utilizadas em rádios, telefones, fax e etc.

Até algumas décadas atrás, a maior parte dos sinais de informação tinha a forma analógica e contínua. Então, vieram os computadores e os circuitos integrados. A proliferação de computadores, que armazenam e processam informação em formas discretas de energia que são medidas em *bits*, gerou forte demanda por equipamentos de transmissão de dados digitais (YOUNG, 2006).

Mudança de chaveamento de frequência (*FSK*, do inglês *Frequency-shift Keying*) é um esquema de modulação de frequência em que a informação digital é transmitida através das mudanças de frequência da onda portadora (PATEL, 2013). Esta estrutura pode ser descrita e implementada tanto em nível de *hardware* como de *software.*

O *FPGA*, sigla em inglês para *Field Programmable Gate Array*, é um circuito integrado com grande poder de processamento e idealizado para ser totalmente programável pelo projetista. A grande vantagem do *FPGA* dentre outros dispositivos se deve a sua alta velocidade de processamento de informação, tendo em vista que em sua estrutura, os dados são todos processados de forma paralela, o que possibilita, por exemplo, uma grande carga de dados sendo processados simultaneamente, o que é justamente requerido em processos de modulação e demodulação. Outra característica que faz o *FPGA* se destacar é a portabilidade, a facilidade de manuseio e programação destes dispositivos é algo de grande valia para a implementação de projetos complexos e que necessitem constantes alterações, por exemplo. Logo, este se apresenta como uma boa alternativa para elaboração de um demodulador digital de frequência.

Sendo assim, este trabalho aborda um estudo a respeito das técnicas de modulação e demodulação, bem como a implementação de um demodulador digital *FSK* (*BFSK*) em *FPGA* por meio de linguagem de descrição de *hardware*, sua análise, simulação e teste de funcionamento.

1.1. JUSTIFICATIVA DO TRABALHO

Com o advento da tecnologia digital, o sinal modulante passou a ser sinais de voz digitalizados e sinais de computadores, isto é, sinais binários cujos conteúdos são zeros (0) e uns (1). A necessidade da modulação para telecomunicações deve-se à necessidade de aumentar a quantidade de canais de transmissão com economia de meios de comunicação (SOARES NETO, 2012).

Por este motivo, surge a exigência de produção de demoduladores digitais a fim de consolidar a comunicação nos sistemas analógico-digitais e formar um dispositivo único - MODEM. O projeto deste trabalho visa justamente conseguir demodular, em um *FPGA*, um sinal cuja informação está modulada em uma onda portadora.

1.2. OBJETIVOS

Neste tópico serão abordados os objetivos, geral e específico, relacionados ao trabalho proposto.

1.2.1. **Objetivo Geral**

Desenvolver um demodulador *BFSK* embarcado em *FPGA*.

1.2.2. **Objetivos Específicos**

- Estudar os conceitos e propriedades das técnicas de modulação e demodulação digital a fim de compreender o seu funcionamento;
- Desenvolver um demodulador *BFSK* embarcado em *FPGA* por meio de uma linguagem de descrição de *hardware*;
- Produzir material bibliográfico que sirva de base para trabalhos futuros.

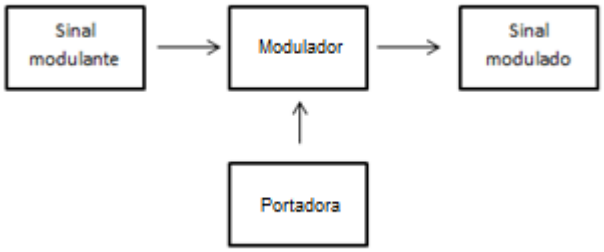
2. FUNDAMENTAÇÃO TEÓRICA

2.1. MODULAÇÃO

Quando duas pessoas tentam se comunicar por meio da voz, por exemplo, estas utilizam as cordas vocais e têm como meio de transmissão o ar para o envio da onda sonora. O papel das cordas vocais, mediante suas vibrações, é causar uma perturbação característica no meio – ar. Tal perturbação chega aos ouvidos do receptor, que por sua vez é transformada em sinal elétrico e processada pelo cérebro. O que fora exemplificado acima é o exemplo mais claro e simples do conceito de modulação.

Modulação é um processo que consiste em se alterar uma característica da onda portadora, proporcionalmente ao sinal modulante (GOMES, 2013). Trazendo para o caso de modulação mostrado anteriormente, a onda portadora seria o ar na garganta do emissor e o sinal modulante seria justamente as vibrações das cordas vocais. De forma geral, o processo de modulação é composto de três sinais: sinal modulante, sinal modulador e sinal modulado.

Figura 1 - Modelo simplificado do processo de modulação.



Fonte: SOARES NETO, 2013.

A definição de modulação passa por um conjunto de ações, em que um sinal modulante é somado a um sinal modulador, resultando um sinal modulado (SOARES NETO, 2013). A representação da forma de modulação mais simples é mostrada na Figura 1. Entretanto, o processo de modulação de sinais é bem mais complexo que o proposto anteriormente.

Com o desenvolvimento da tecnologia, o sinal modulador passou a ter uma frequência muito mais elevada que a frequência do sinal modulante, criando vários outros sistemas de comunicação, como as ondas portadoras (transmissão de alta frequência sobre fios), os

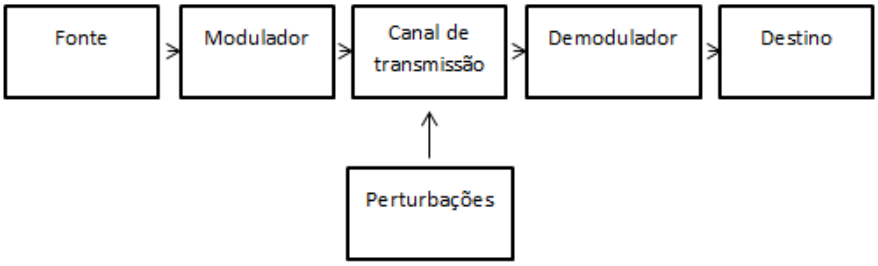
sistemas de rádio enlace, satélite e fibra ótica (SOARES NETO, 2013). Este aumento de frequência é devido principalmente à procura de sistemas cada vez mais eficientes e que se encaixem as suas devidas especificações.

De modo geral, o processo de modulação consiste em fazer um dos parâmetros de uma onda portadora senoidal, por exemplo, $p(t) = E_o * \cos(2\pi f_o t + \phi)$ variar acompanhando o sinal da informação *x(t)* - voz, imagem, dados, etc, - que se quer transmitir (o sinal modulante). Os parâmetros da onda portadora que podem ser alterados pelo sinal *x(t)* são sua amplitude (E_o) e seu ângulo de fase (ϕ), resultando nos processos de modulação de amplitude e modulação angular, respectivamente (CARVALHO, 2009). Sendo a modulação angular subdividida em modulação de frequência e modulação de fase.

2.1.1. **Demodulação**

A demodulação é o processo inverso ao da modulação. Pode-se dizer que o sinal modulado é convertido novamente no sinal original (SOARES NETO, 2013). Ou seja, se retira a informação que antes estava codificada junto a onda portadora para que fosse enviada. Este processo completa o ciclo modulação-demodulação. A Figura 2 representa o seu esquema geral.

Figura 2 – Esquema completo da modulação e demodulação da informação.



Fonte: SOARES NETO, 2013.

No processo de demodulação de sinais, é necessária a detecção do sinal por parte do demodulador, e para esta tarefa existem alguns métodos. Os principais métodos são por detecção não coerente, detecção diferencialmente coerente e detecção coerente. Na detecção não coerente, o demodulador necessita estimar certos parâmetros do sinal recebido, como a fase e frequência, por exemplo. A detecção diferencialmente coerente ocorre quando o

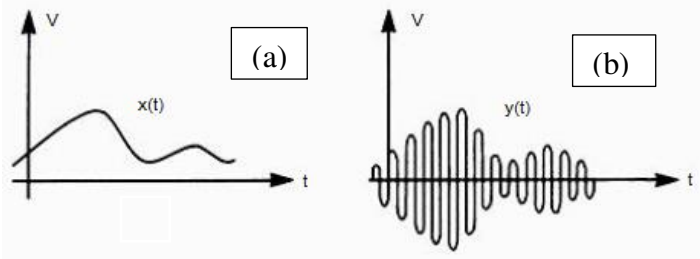
módulo demodulante compara a fase do sinal recebido com um valor prévio. No processo de detecção coerente, existem, tanto no modulador como no demodulador, osciladores previamente sintonizados, de maneira que o sinal é detectado no demodulador a partir da comparação entre as fases do sinal recebido e do sinal gerado pelo oscilador próprio do demodulador (TONGUZ; WAGNER, 1991). As eficiências destes processos diferem entre si devido ao próprio mecanismo de detecção de sinal e o tipo de modulação que fora aplicado ao sinal portador.

Nunca o processo de modulação e demodulação é feito com cem por cento de eficiência, pois existem sempre falhas, tanto nos sistemas moduladores como nos demoduladores, bem como as perturbações que o sinal sofre ao ser transmitido pelo meio (SOARES NETO, 2013). A busca pela eficiência destes processos gera diversos estudos e pesquisas a respeito deste tema, resultando em sistemas de modulação cada vez mais complexos e sofisticados com altos índices de desempenho.

2.1.2. **Modulação de Amplitude**

Tendo-se uma onda portadora do tipo senoidal $p(t) = Eo * \cos(2\pi fot + \phi)$, modulação de amplitude é, essencialmente, o processo de produto do sinal de informação *x(t)* por um sinal senoidal *p(t)* – a onda portadora, com o objetivo de deslocar o espectro para um intervalo de frequência adequado a uma determinada aplicação. Fisicamente, efetua-se esse processo em um modulador de produto, um circuito que recebe os sinais *p(t)* e *x(t)*, e produz um sinal elétrico de saída $y(t) = KM * x(t) * \cos(2\pi fot)$, onde KM é a constante de modulação que caracteriza este circuito (CARVALHO, 2009). Este produto fará com que a amplitude da onda portadora varie proporcionalmente ao sinal da informação (*x(t)*). A Figura 3 representa este processo.

Figura 3 - Modulação de Amplitude.



Fonte: YOUNG, 2006.

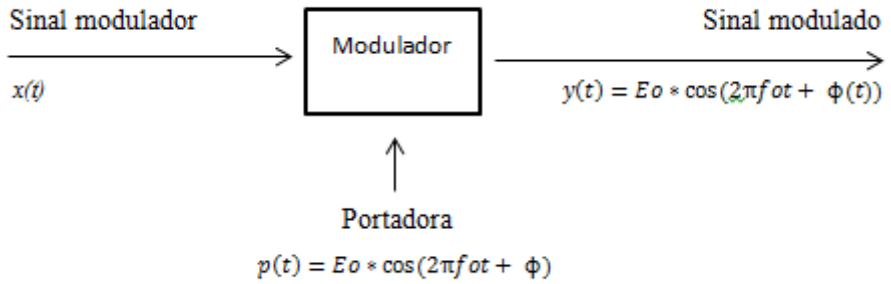
A modulação de amplitude (*AM* – sigla do inglês para *Amplitude Modulation*) é dividida em diversos métodos, cada um deles tendo alguma particularidade no processo de composição do sinal, entre elas: *AM-DSB/TC, AM-DSB/SC, AM-SSB/TC, AM-SSB/SC*.

2.1.3. **Modulação de Frequência**

Antes de definir o mecanismo de modulação de frequência, é necessário introduzir o conceito de modulação angular.

Denomina-se modulação angular o sistema de modulação de uma portadora senoidal em que a amplitude do sinal modulado se mantém constante e seu ângulo varia, acompanhando o sinal modulador (CARVALHO, 2009). Como mostrado na Figura 4.

Figura 4 – Modulação Angular.

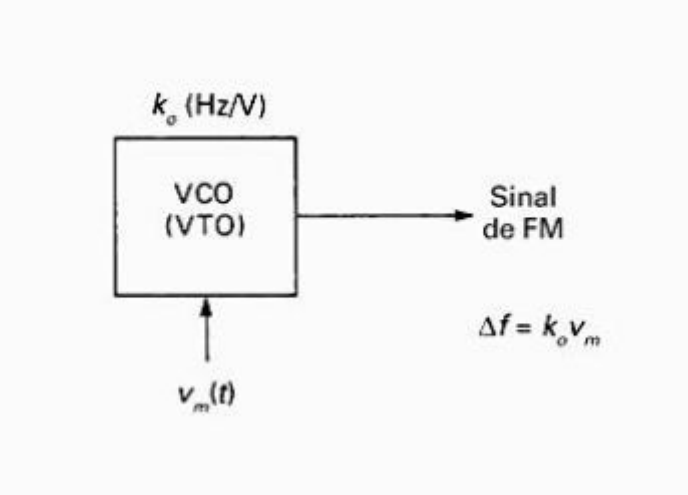


Fonte: CARVALHO, 2009.

Como já mencionado anteriormente, a modulação angular pode ser desmembrada em outros dois tipos de modulação, a modulação de fase e a modulação de frequência.

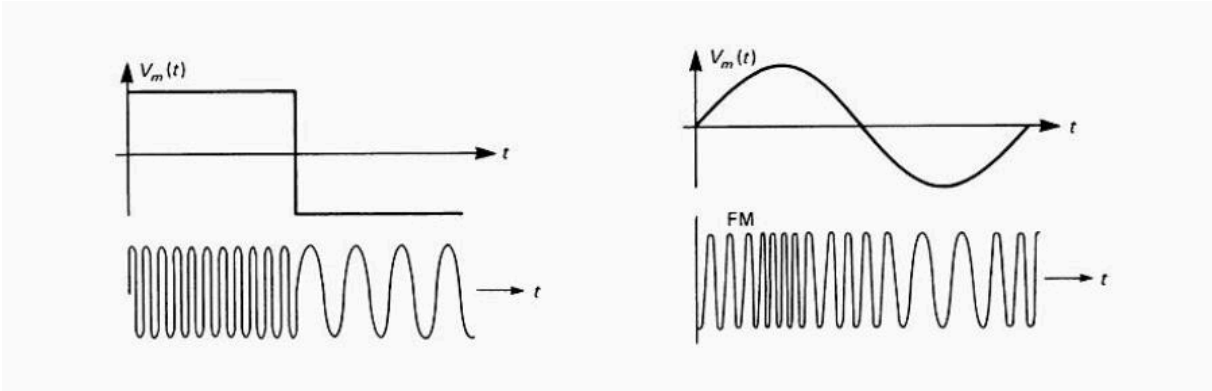
Na modulação de frequência (*FM*) temos que os parâmetros de amplitude de onda e fase permanecem constantes e inalterados, ocorrendo apenas mudanças na frequência do sinal. Um sinal com frequência modulada é qualquer sinal periódico cuja frequência instantânea *f* seja desviada de um valor médio f_c por um sinal de informação *m(t)* (YOUNG, 2006). A modulação de frequência é realizada por um módulo comumente chamado de *VCO* (sigla do inglês para *Voltage-controlled Oscillator)*, Este módulo tem uma constante de proporcionalidade k_o que indica a amplitude do desvio de frequência Δf que será produzido em decorrência do sinal de tensão *m(t)*. A figura 5 apresenta este módulo e a Figura 6 mostra o seu funcionamento com duas formas de onda.

Figura 5 – Bloco *VCO*.



Fonte: YOUNG, 2006.

Figura 6 – Funcionamento do VCO para m(t) quadrada e senoidal.

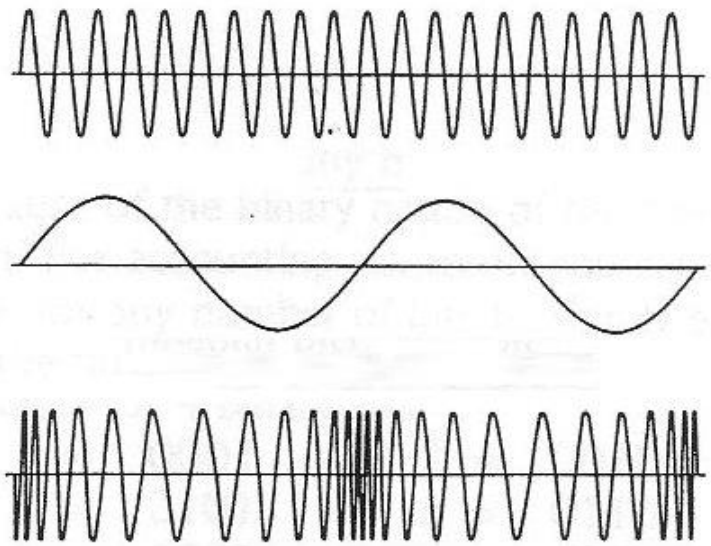


Fonte: YOUNG, 2006.

2.1.4. **Modulação de Fase**

Conhecida pela sigla *PM* (do inglês, *Phase Modulation*), a modulação de fase é o processo de modulação angular no qual o desvio de fase é diretamente proporcional ao sinal modulador (CARVALHO, 2009). Existe a constante de proporcionalidade *Kp* que indica a sensibilidade do modulador *PM*. O sinal modulado em fase pode ser descrito como $y(t) = Eo * \cos(2\pi fot + Kp * x(t))$, onde o termo *Kp*x(t)* define a variação da fase do sinal modulado. A figura 7 exemplifica a modulação de fase com sinal continuo no tempo junto a uma portadora senoidal.

Figura 7 – Modulação de fase com portadora senoidal.



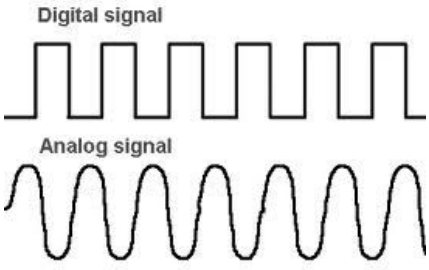
Fonte: YOUNG, 2006.

2.2. MODULAÇÃO DIGITAL

2.2.1. **Fundamentos da Conversão Analógico Digital**

O processo de transformação de um sinal continuo ao longo do tempo, dito analógico, num sinal com valores discretos no domínio do tempo, chamado de sinal digital, é conhecido como conversão analógico digital. Os conversores analógico-digital (A/D) são equipamentos que convertem o sinal analógico em digital (ALSINA et al., 1999). Na figura 8, têm-se dois exemplos de sinal, o primeiro com valores discretos ao longo do tempo – sinal digital, e o segundo com valores contínuos no domínio do tempo – sinal analógico.

Figura 8 – Sinal digital e sinal analógico.



Fonte: http://ricardosilvadisciplinaderedes.blogspot.com.br/2010/10/transmissao-de-sinais-analogicos-e.html.

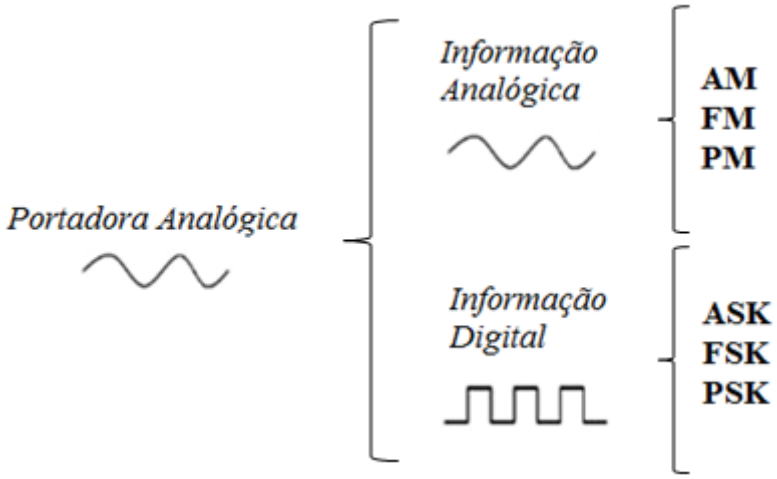
2.2.2. **Fundamentos de Modulação Digital**

O conceito de modulação até agora abordado (item 2.1.) consistia em alterar alguma característica de uma onda portadora senoidal proporcionalmente a um sinal de informação contínuo ao longo do tempo, esta alteração, que por sua vez poderia ocorrer na amplitude, na frequência ou na fase da portadora, servira para facilitar o processo de transmissão do sinal de informação de um ponto para outro do enlace de comunicação. Como o sinal de informação era continuo no domínio do tempo, consideramos este um sinal analógico, logo, o mecanismo de modulação também era dito analógico.

Entretanto, para um sinal de informação em que seus valores são discretos ao longo do tempo, variando de zero para um ou vice-versa, por exemplo, temos então um sinal digital.

A modulação do sinal de informação digital com uma portadora é dita modulação digital. Os processos básicos de modulação de uma portadora senoidal por um sinal modulador digital são conhecidos por chaveamento de amplitude, de frequência e de *fase* – identificados, respectivamente, pelas siglas *ASK*, *FSK* e *PSK* (do inglês *amplitude, frequency, phase shift keying*) (CARVALHO, 2009). Na figura 9 estão representados os principais tipos de modulação com portadora senoidal, seja a informação digital ou analógica.

Figura 9 - Principais tipos de modulação.



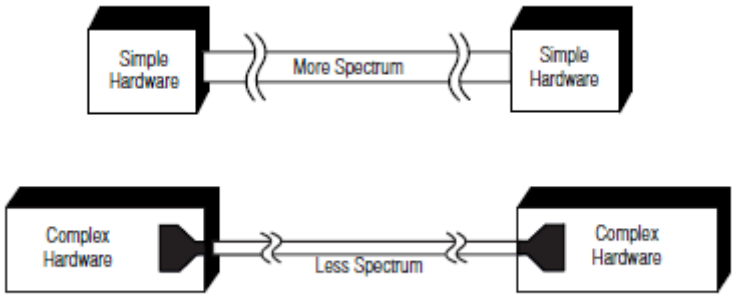
Fonte: Autoria própria.

A migração para a modulação digital gera diversas mudanças nos sistemas de comunicação, aumento da capacidade de informação, maior compatibilidade com dispositivos que trabalham com dados digitais, maior segurança de dados, melhor qualidade de

comunicação, e maior velocidade nos sistemas disponíveis. Sistemas de comunicação com *hardwares* simples transmitem e recebem informação utilizando um grande espectro de banda, já sistemas que detêm uma arquitetura mais complexa (com modulação digital, por exemplo), conseguem realizar esta tarefa com uma largura de banda muito menor, resultando assim em uma elevada eficiência espectral (AGILENT TECHNOLOGIES, 2001). A figura 10 mostra isto.

Para os sistemas de demodulação digital, seguem os mesmos princípios da demodulação analógica, diferenciando-se pelos conceitos de modulação digital. Em vez de recuperar um sinal analógico de informação, temos na saída do demodulador um sinal digital de informação.

Figura 10 – Esquema relacionando a complexidade de hardware do sistema de comunicação com a densidade de informação no espectro de transmissão.



Fonte: AGILENT TECHNOLOGIES, 2001.

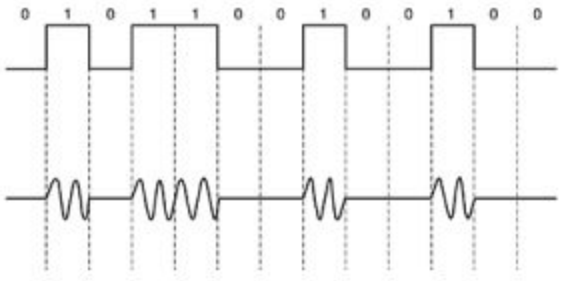
2.2.3. **Modulação *ASK***

A definição de modulação *ASK* (sigla do inglês para *Amplitude Shift Keying*) consiste na variação de amplitude da portadora senoidal, realizada mediante o estado lógico do sinal modulador (SOARES NETO, 2013).

Em um modulador de produto com constante de modulação *Km*, o sinal modulado *ASK* é: $e(t) = Km * m(t) * \cos(2\pi f_o * t)$. O sinal digital binário x(t) é convertido em um sinal multinível *m(t)*, gerando níveis de amplitude diferentes na portadora (CARVALHO, 2009). A modulação *M-ASK* tem $M = 2^n$ níveis de tensão associados a cada combinação de *bits*. Sendo o sinal de informação binário de um *bit*, tem-se dois níveis de tensão, logo, tem-se

um sinal modulado *2-ASK* ou *OOK* (do inglês, *On-off Keying)*. Na figura 10 está representada as formas de onda para este tipo de modulação.

Figura 11 – Formas de onda da modulação *OOK* (On-off Keying).



Fonte: http://www.teleco.com.br/tutoriais/tutorialwifimanaus1/pagina_3.asp.

2.2.4. **Modulação *FSK***

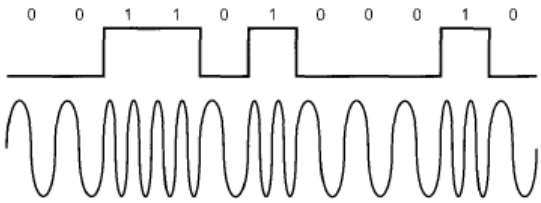
Com um sinal digital de *M* níveis aplicado a um modulador de frequência e portadora senoidal de frequência *Fc*, tem-se na saída do modulador um sinal modulado com amplitude constante e *M* valores discretos de frequência (CARVALHO, 2009). Sendo dois níveis de tensão, têm-se dois valores discretos de frequência no sinal modulado *BFSK* (do inglês, *Binary Frequency Shift Keying*), como mostrado na figura 12.

$$e(t) = \begin{cases} E_0 * \cos(2\pi * f_1 * t) \\ E_0 * \cos(2\pi * f_2 * t) \end{cases}$$

Pode-se observar nas formas de onda na figura 12 que a frequência da portadora têm variações de acordo com o nível lógico (1 ou 0) do sinal modulante. Quando nível lógico 1, a portadora assume uma frequência mais alta, já quando em nível lógico 0, a frequência da portadora tem um valor menor.

Uma das maneiras de se demodular um sinal *BFSK* é através de um *PLL* (sigla do inglês para *Phase Locked Loop*). *PLL* é um circuito que pode ser implementado tanto de maneira analógica (*PLL*) como digital (*DPLL*), este detém diversas funcionalidades de processamento de sinal, sendo uma delas o de recuperação de sinais (sinal da portadora e sinal de informação). A estrutura e funcionalidades de um *PLL* serão explicadas mais a frente.

Figura 12 – Modulação BFSK (2-FSK).



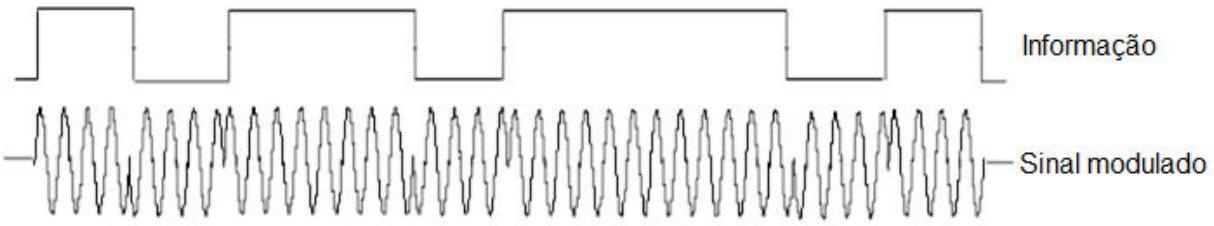
Fonte: http://www.mibqyyo.com/pt-artigos/2014/08/04/comunicacoes-via-radio-ii-tipos-de-modulacao/.

2.2.5. **Modulação *PSK***

A modulação *PSK* (sigla do inglês para *Phase Shift Keying)* tem base na variação da fase da portadora senoidal mediante as variações do estado lógico do sinal modulador. A frequência da portadora não se altera e a informação digital é transmitida em fase com a portadora (SOARES NETO, 2013).

Para um sinal de informação com dois níveis lógicos possíveis (digital – 1 e 0) tem-se na saída do modulador *PSK* um sinal senoidal com variações abruptas de fase (180º), como mostrado na figura 13.

Figura 13 – Modulação BPSK.



Fonte: Autor própria.

Observa-se que a cada mudança de nível lógico, ocorre uma mudança 180º na fase do sinal senoidal.

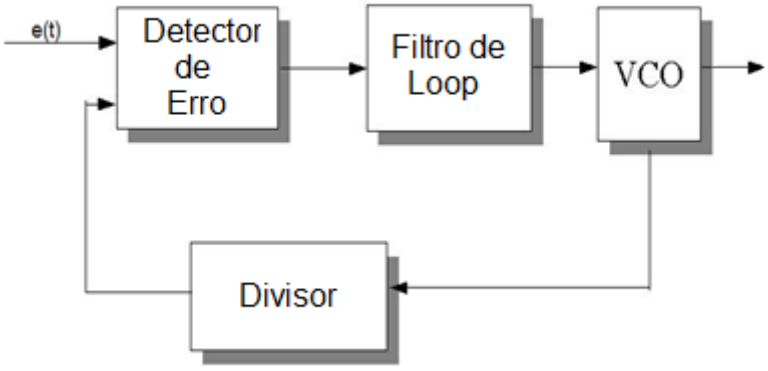
Outro tipo bastante comum de modulação de fase é a *QPSK* (4-*PSK*), onde a informação pode ter 4 níveis lógicos distintos (00, 01, 10 e 11), deste forma, cada mudança de fase da portadora corresponde a dois valores de bits.

2.3. *PHASE LOCKED LOOP*

Phase Locked Loop (*PLL)* é um sistema que trava a fase ou frequência de um sinal de referência de entrada. *PLL*'s são comumente utilizados em computadores, rádios e sistemas de telecomunicação, onde é necessário estabilizar um sinal gerado ou detectar sinais (ELSEROUGI, 2006). Como o próprio nome já indica, o objetivo de um *PLL* é gerar um sinal que esteja na mesma fase e/ou frequência do sinal de referência, mesmo este sofrendo variações, seja por conta de projeto ou devido interferências.

O sistema funciona inicialmente a partir da comparação de um sinal de entrada com um sinal gerado no próprio *PLL*, este sinal advém de um oscilador próprio do sistema e que seu sinal de saída retroalimenta o *PLL* (BEST, 2007). A figura 14 mostra um esquema básico de um circuito *PLL.*

Figura 14 – PLL básico.



Fonte: BEST, 2007.

O primeiro bloco (*PFD*, sigla do inglês para *Phase-Frequency Detector*) é o detector de erro de fase/frequência. Ele é alimentado pelo sinal que se deseja processar no *PLL* e por um sinal de referência que é gerado pelo próprio sistema. Este bloco tem como função indicar em sua saída quando houver alguma diferença entre os sinais de entrada. Dependendo da aplicação, este bloco pode ser projetado para detectar apenas diferenças de fase, de frequência, ou de ambos. Havendo alguma diferença entre os sinais de entrada, o *PFD* produzirá um valor de tensão proporcional a diferença detectada por ele.

O segundo bloco do *PLL* consiste em um *Loop Filter*, ou Filtro de Laço. Este bloco geralmente é implementado como um filtro passa-baixa. Ele recebe a variação de tensão em

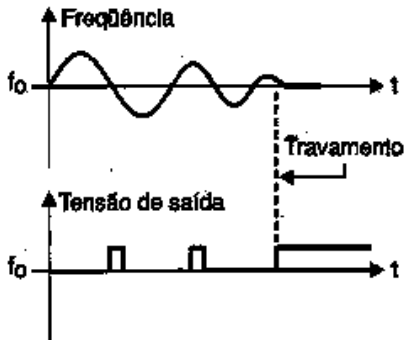
função do erro diferencial gerado pelo detector de erro, e filtra a tensão, tendo em sua saída em um sinal *DC* proporcional a esta diferença de tensão.

O *VCO* (sigla em inglês para *Voltage-Controlled Oscillator*), ou oscilador controlado por tensão, é um dos blocos mais importantes de um *PLL*, pois tem a função de gerar um sinal variante no tempo (oscilação) de frequência proporcional ao seu sinal de entrada advindo do *Loop Filter*. O *VCO* é um oscilador que retroalimenta o *PLL*.

Por último, temos o Divisor de Frequência. Nem sempre utilizado nos projetos de *PLL*, ele tem a função de dividir a frequência do sinal gerado pelo *VCO*. Normalmente, o Divisor é adotado a fim de melhorar o sincronismo entre os sinais que alimentam o *PFD* (sinal de entrada e o sinal de referência), gerando submúltiplos da frequência do oscilador.

De maneira simples, o *PLL* funciona da seguinte forma: aplicando-se um sinal de entrada com uma frequência *f* e tendo o *VCO* um sinal inicial de frequência *fo*, o *PFD* irá gerar um valor de tensão proporcional à diferença de frequências (*f-fo*), que passará pelo filtro (*Loop Filter*) e irá, em seguida, controlar o oscilador. O *VCO* irá gerar um sinal de frequência cada vez mais semelhante a *f*, fazendo com que a diferença entre as frequências seja cada vez menor. No instante em que as duas frequências dos sinais se igualarem, o sinal produzido pelo detector de erro será zerado e a tensão que controla o *VCO* se estabilizará, fazendo este travar na frequência semelhante a do sinal de entrada do *PLL*. Os gráficos da figura 15 mostram isso.

Figura 15 – Travamento do PLL.



Fonte: http://www.newtoncbraga.com.br/index.php/como-funciona/624-como-funciona-o-pll-art058.

O *PLL* tem a capacidade de recuperar a frequência exata de um sinal (ou componente espectral) de entrada – saída do *VCO* – o que terá aplicação imediata na recuperação de portadoras (MILAGRE JÚNIOR, 2001). Os módulos que compõem um *PLL* podem ser projetados para ser tanto analógicos como digitais, dependendo da aplicação.

2.3.1. *All Digital Phase Locked Loop*

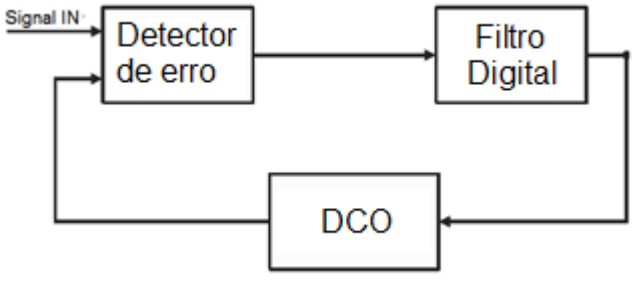
Existem diversos tipos de sistemas *PLL*, onde eles diferem, além da aplicação, pela forma com que os módulos que constituem o sistema são projetados. O *Phase Locked Loop* foi proposto inicialmente de forma analógica, entretanto, com o avanço tecnológico e a necessidade de maiores níveis de complexidade de *hardware* e carga de processamento, surgiram variações para este sistema, nos quais alguns módulos ou todos foram projetados para trabalhar de maneira digital, ou seja, com valores discretos.

Os sistemas *PLL*'s podem ser divididos em:

- *PLL* analógico: todos os módulos implementados analogicamente.
- *DPLL* (*Digital PLL*): boa parte dos módulos são analógicos, porém alguns deles são implementados com blocos digitais.
- *ADPLL* (*All Digital PLL*): todos módulos são implementados utilizando blocos digitais.

O *PLL* analógico ainda é bastante utilizado, porém o sistema digital (*DPLL*) vem atraindo bastante atenção pelas suas significantes vantagens em circuitos digitais sobre o modelo analógico de *PLL*. Estas vantagens incluem superioridade em desempenho, velocidade, confiabilidade e redução de tamanho e custo (AL-ARAJI; HUSSAIN; AL-QUTAYRI, 2006).

Figura 16 – Esquema de um ADPLL.



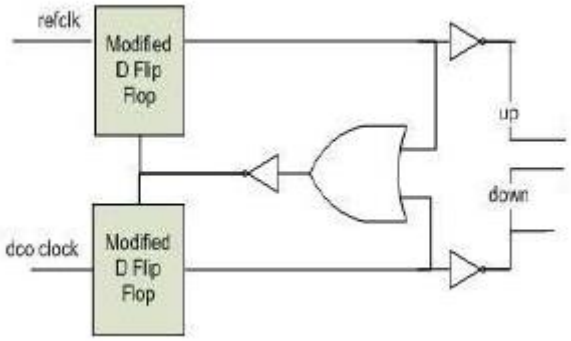
Fonte: Autor própria.

No esquema mostrado na figura 16 tem-se um exemplo geral para um sistema *ADPLL*. O módulo detector de erro (*Error Detector*) pode ser projetado de diversas formas, como a partir de uma Porta XOR (*Exclusive – OR*) (figura 17) ou por *Phase Frequency Error Detector* (*PFD*) (figura 18). Os dois tipos são sensíveis à diferença entre os sinais de entrada.

Figura 17 – Porta XOR.

Fonte: Autoria Própria.

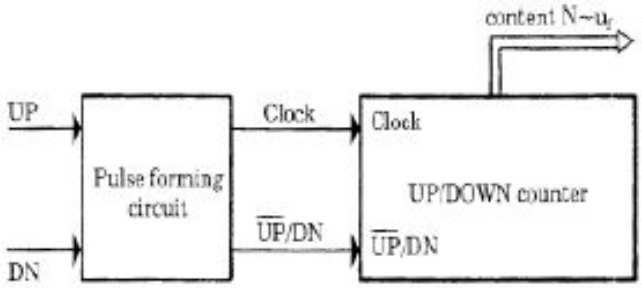
Figura 18 – PFD.



Fonte: (LATA; KUMAR, 2013).

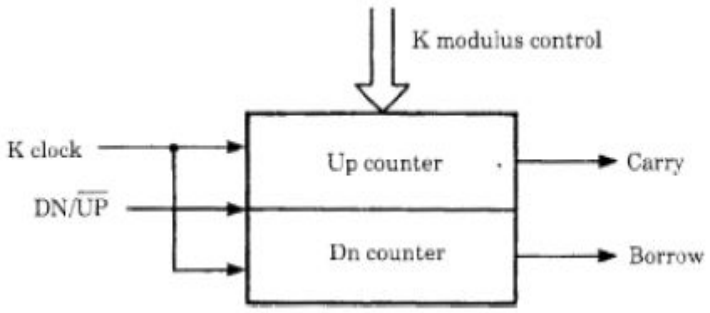
O *Digital Filter* (Filtro digital) é um simples filtro digital que pode ter dois esquemas básicos: *Up/Down Counter* (figura 19) ou *K-Counter* (figura 20).

Figura 19 – Up/Down Counter.



Fonte: (LATA; KUMAR, 2013).

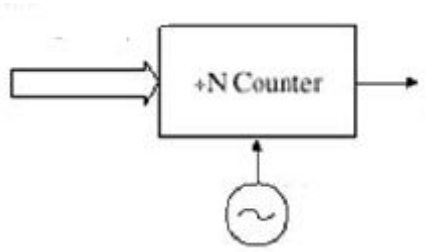
Figura 20 – *K-Counter*.



Fonte: (LATA; KUMAR, 2013).

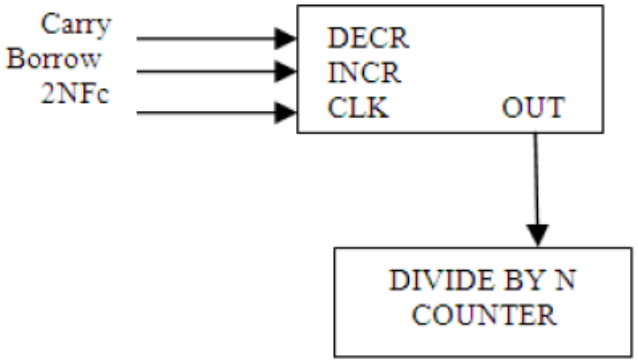
O ultimo bloco, *DCO* (sigla do inglês para *Digital Controlled Oscillator*) segue a mesma função do *Voltage Controlled Oscilator*, porém trabalho com valores digitais para controlar a frequência do seu oscilador interno. Ele pode ser projetado de duas maneiras: *Div By N Counter* (figura 21) ou *I/D Counter* (figura 22).

Figura 21 – *Div By N Counter*.



Autor: (LATA; KUMAR, 2013).

Figura 22 – *Increment/Decrement Counter*.



Fonte: (LATA; KUMAR, 2013).

2.4. *FIELD-PROGRAMMABLE GATE ARRAY*

2.4.1. **Surgimento**

Com o desenvolvimento constante dos recursos de concepção de circuitos integrados, tornou-se possível o aparecimento de dispositivos com lógica programável, isto é, dispositivos que pudessem ser programados pelos próprios usuários, possibilitando o desenvolvimento de circuitos mais complexos sem a necessidade de manipulação direta do silício (KILTS, 2007).

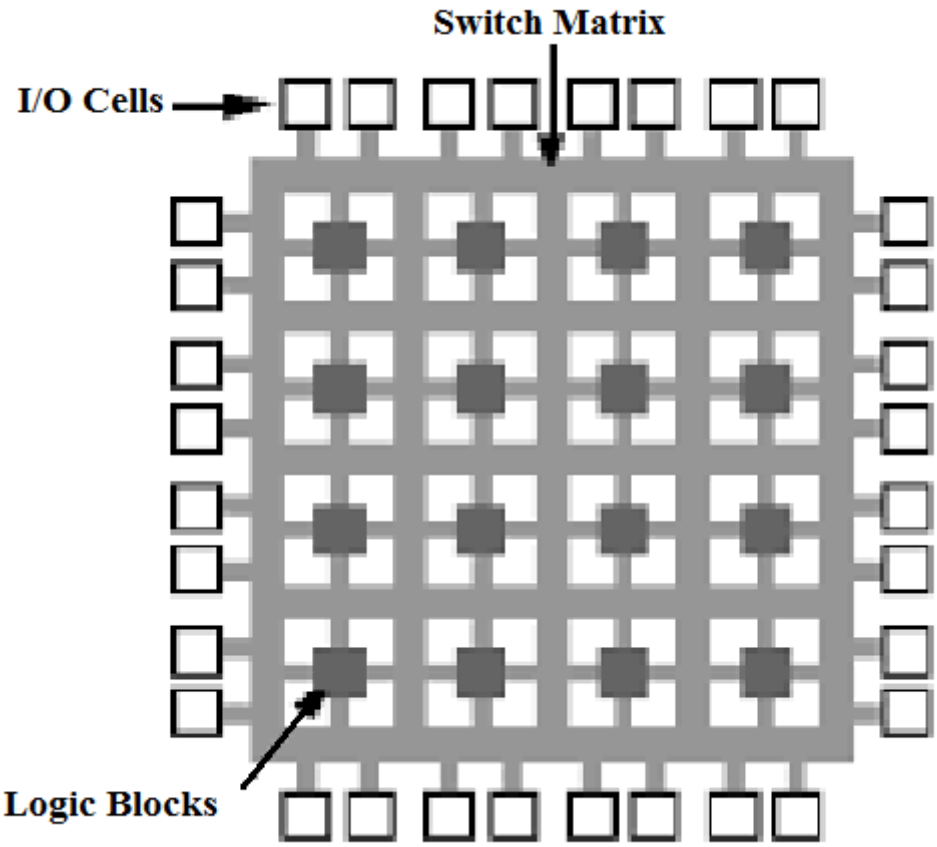
FPGA (do inglês, *Field-Programmable Gate Array*) ou Arranjo de Portas Programáveis em Campo, é um dispositivo (circuito integrado – *CI*) de lógica programável (COSTA, 2014). Ross Freeman inventou o primeiro *FPGA* em 1985. Ao contrário dos processadores comuns, os *FPGA*'s trabalham de forma paralela. Diferentes operações de processamento não competem pelos mesmos recursos do *CI*, cada tarefa é destinada para uma parte específica do *FPGA* e assim realizada, não sofrendo qualquer influência dos outros processos que estão sendo realizados simultaneamente. Como resultado, tem-se uma alta capacidade e velocidade de processamento de tarefas simultâneas. Outra característica marcante nestes dispositivos é que a lógica que é programada é implementada diretamente a nível de *hardware* e não executada por um sistema operacional ou outros softwares, como é o caso dos sistemas baseados em processadores (COSTA, 2014).

2.4.2. **Funcionamento**

A estrutura interna de um *FPGA*, apresentado na figura 23, é constituída basicamente de um conjunto de Blocos Lógicos, ou *CLB*'s (sigla do inglês para *Configurable Logic Blocks*), distribuídos em forma de matriz pelo *CI*, estes blocos lógicos são interligados por um sistema de Roteamento. Existem ainda os Blocos de Entrada e Saída, ou *I/O Blocks*, que fazem a conexão do *FPGA* com o mundo exterior.

As funções programadas pelo usuário são implementadas nos Blocos Lógicos, esses são constituídos por *flip-flops*, registradores, *ULA*'s e *Lookup Tables* (COSTA, 2014). O Roteamento é feito de maneira que os Blocos Lógicos possam se comunicar entre si e entre os Blocos de Entrada e Saída da forma mais rápida e eficiente possível, para isso existem vários níveis de conexão e as chamadas *Switch Matrix*, que são chaves de conexão que servem para aperfeiçoar o processo de Roteamento.

Figura 23 – Estrutura interna de um *FPGA*.



Fonte: Autoria Própria.

Com a finalidade de se programar em *FPGA*, surgiram as ferramentas de desenvolvimento, estas tinham como objetivo facilitar a interação entre o projetista e a lógica interna do *FPGA*. Após o desenvolvimento da descrição de *hardware*, a ferramenta compila esta lógica em um arquivo de configuração que contém todas as informações para programar cada bloco interno do *FPGA*, bem como a maneira que estes serão interligados a fim de se atingir o objetivo do projeto.

Ao longo dos anos, as ferramentas de desenvolvimento se tornaram mais complexas e cada vez mais eficientes, acompanhando isto, as *HDL*'s (sigla do inglês para *Hardware Description Languages* ou Linguagens de Descrição de *Hardware*) evoluíram de linguagens simples para projetos de algoritmo para execução em *chips FPGA*. Entre as *HDL*'s que se destacam, pode-se citar *VHDL* e *Verilog*. Estas linguagens reúnem características de outras linguagens de programação, como Linguagem C, porém detêm aspectos particulares que se adequam à especificações de projeto em *FPGA*, como suas sintaxes que têm que mapear

sinais de entrada advindos dos Blocos de Entrada e Saída, os leva para os Blocos Lógicos e refazem o caminho oposto.

É comum a utilização dos chamados *Kits* de Desenvolvimento, onde está contido um *FPGA* junto a periféricos ligados entre si em uma placa. Este conjunto, após programado pelo projetista, realiza uma função específica e é uma boa alternativa nos processos de prototipagem de projetos mais complexos.

3. DESENVOLVIMENTO DO DEMODULADOR BFSK

Neste trabalho, foi desenvolvido um *ADPLL* para a construção do demodulador em forma de bloco lógico digital. A seguir, será mostrado mais afundo o funcionamento de cada módulo que compõem o demodulador *BFSK* desenvolvido.

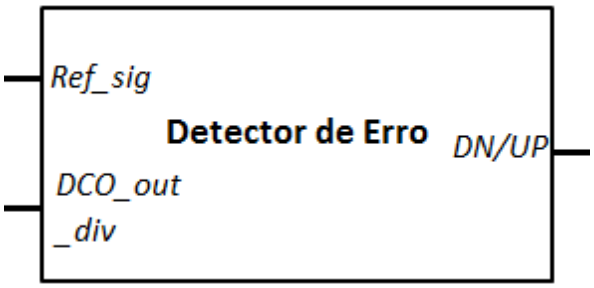
3.1. ARQUITETURA

3.1.1. Detector de Erro

O primeiro módulo, o Detector de Erro, do *ADPLL*, pode ser implementado de diversas maneiras, seja através de uma porta lógica *XOR*, um detector de erro de frequência e fase por meio da combinação de *flip-flop*'s e portas lógicas básicas (*AND*, *OR*, *NOT*), ou através de um *ECPD* (sigla do inglês para *Edge Controled Phase Detector*), que fora o escolhido para ser utilizado como o detector de erro para o *ADPLL* (figura 24).

O *ECPD* tem como entradas o sinal *FSK* modulado (representado por *Ref_sig*) e o sinal de realimentação advindo do *DCO* e do Divisor de Frequência (representado por *DCO_out_div*). Ele atua como uma espécie de *flip-flop JK*, produzindo uma saída (*DN/UP*) em nível lógico alto (1) na transição de nível alto para nível baixo do sinal *DCO_out_div*, e de maneira contrária, uma saída em nível lógico baixo (0) na transição de nível baixo pra nível alto do sinal de entrada *Ref_sig*.

Figura 24 - Módulo Detector de Erro



Fonte: Autoria própria.

Por meio deste esquema, o *ECPD* consegue determinar uma diferença de fase (erro) entre o sinal modulado e o de realimentação, podendo captar erros de até 50% *duty* de ciclos

de frequência, ao contrário de um Detector de Erro produzido a partir de uma porta *XOR*, que alcança no máximo 25%.

3.1.2. Filtro de *Loop*

O filtro de *Loop,* ou *Loop Filter*, é o segundo módulo do *ADPLL*, ele tem a função de receber o sinal gerado pelo Detector de Erro (*DN/UP*), processar este na forma de uma contagem a partir de um sinal de *clock* chamado de *K_clk*, e gerar como saída dois sinais, *Carry* e *Borrow* (figura 25).

Figura 25 - Módulo Filtro de *Loop*

Fonte: Autoria própria.

Este bloco é basicamente a junção de dois contadores digitais com um módulo *K* cada, definido como um parâmetro de entrada do *ADPLL* que se relaciona com a faixa de frequência que o circuito digital irá trabalhar. Mais a frente, será discutida a escolha do *K* e entre outros parâmetros para o correto funcionamento do demodulador.

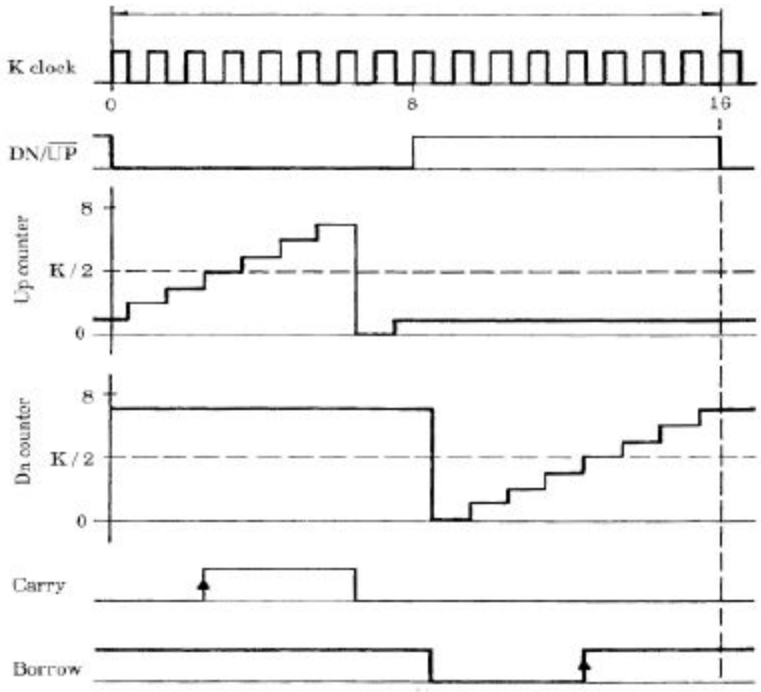
Também chamado de *K-Counter*, o Filtro de *Loop* atua a partir de duas contagens que terão seu desenvolvimento relacionado com o nível lógico do sinal *UP/DN*.

A primeira das contagens, *UP_Counter*, funciona da seguinte forma, enquanto *DN/UP* estiver em nível lógico baixo (0), o contador atua somando +1 à sua contagem a cada borda de subida do *K_clk*. A partir do momento que a contagem atingir o valor de *K/2*, o Filtro de *Loop* gera um nível lógico alto (1) na saída *Carry*. Quando a contagem atinge o limite *K-1*, esta é reiniciada e continua seu desenvolvimento.

A segunda contagem, *DN_Counter*, funciona a partir do instante que o sinal *DN/UP* estiver em nível lógico alto (1), este ocorrendo, o contador soma +1 à sua contagem. Quando a contagem estiver igual ou superior a *K/2*, o Filtro de *Loop* gera na sua saída um valor alto

(1) para *Borrow*. Analogamente ao *UP_Counter*, a contagem quando atinge *K-1*, é reiniciada e prossegue seu funcionamento normalmente. A figura 26 demonstra o correto funcionamento do *K_Counter*.

Figura 26 - Exemplo de funcionamento do *K_Counter*



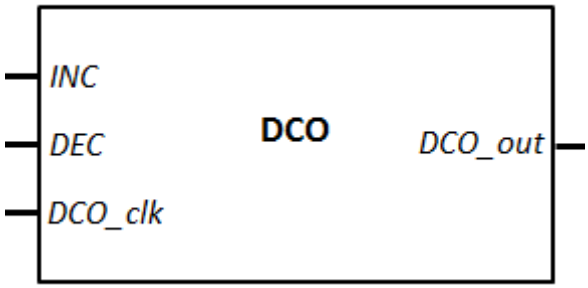
Fonte: (LATA; KUMAR, 2013).

3.1.3. Oscilador Controlado Digitalmente

O Oscilador Controlado Digitalmente, ou *DCO*, é basicamente um oscilador. O módulo *DCO* tem como entradas os sinais gerados pelo Filtro de *Loop*, o sinal *Carry* é associado internamente à entrada *INC*, e o sinal *Borrow* associado à entrada *DEC*. O *DCO* tem também como entrada um sinal de *clock* de alta frequência, o oscilador propriamente dito. O módulo *DCO* é apresentado na figura 27.

O funcionamento do *DCO* consiste na mudança de frequência do oscilador interno com base nos sinais de entrada *INC* e *DEC*. Estes sinais irão definir se a frequência do sinal de saída do *DCO*, *DCO_out*, irá aumentar ou diminuir. Este módulo atua de três maneiras diferentes, que serão agora explanadas.

Figura 27 - Módulo *DCO*.

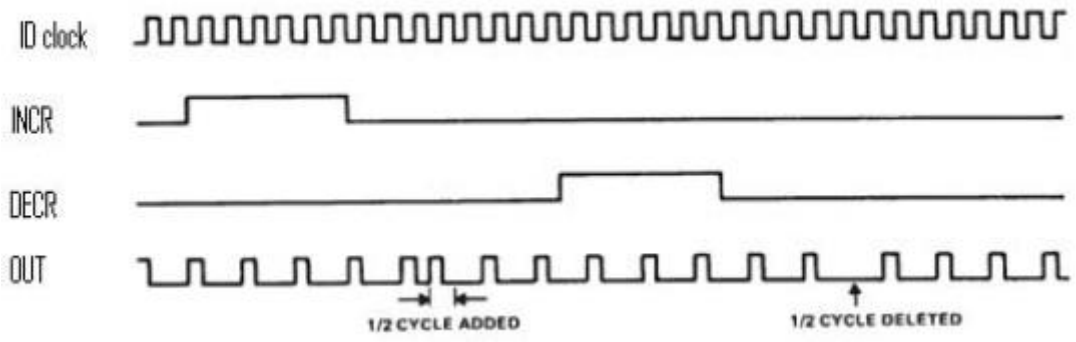


Fonte: Autoria própria.

O primeiro modo de funcionamento do *DCO* ocorre quando ambos os sinais *INC* e *DEC* estão em nível baixo (0). Havendo este estado, a lógica interna do *DCO* fará com que a sua saída (*DCO_out*) seja a igual ao *clock* de entrada (*DCO_clk*), porém com a frequência dividida por dois.

Para quê ocorra o segundo modo, é necessário que o nível lógico de *DEC* esteja em nível alto (1). Nestas circunstâncias, *DCO_out* recebe o *clock* de entrada com a frequência dividida por dois porém com meio ciclo de *clock* retirado, resultando assim numa variação negativa na frequência do sinal de saída.

Figura 28 - Funcionamento do módulo *DCO*.



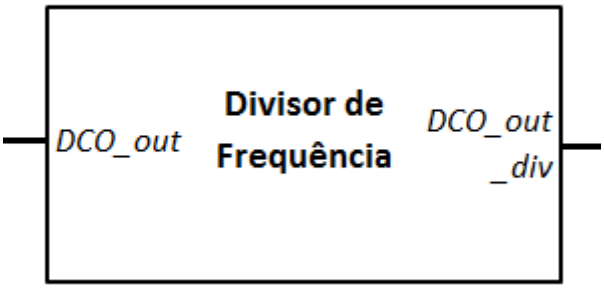
Fonte: (LATA; KUMAR, 2013).

Quando se tem o sinal de entrada *INC* em estado lógico alto (1), ocorre o terceiro modo de funcionamento. *DCO_out* recebe o *clock* de entrada com a frequência dividida por dois, mas com a adição de meio pulso de *clock*, resultando numa variação positiva na

frequência do sinal de saída (*DCO_out*). A figura 28 demonstra as formas de onda resultando do funcionamento do *DCO*.

3.1.4. Divisor de Frequência

Figura 29 - Módulo Divisor de Frequência.



Fonte: Autoria própria.

Em decorrência da utilização de um sinal de *clock* de alta frequência no *DCO*, é necessário que esta frequência seja reduzida para a faixa de atuação do *ADPLL* para quê seja possível a comparação dos sinais (sinal de referência e sinal do *DCO*) no Detector de Erro e este funcione de modo correto. Para isto, é necessária a utilização de um Divisor de Frequência. Com um módulo *N*, definido como um parâmetro de entrada do demodulador, o Divisor de Frequência, como o próprio nome já explica, irá dividir a frequência do sinal de entrada *DCO_out* e gerar na sua saída um sinal com frequência mais baixa *DCO_out_div*. O valor de *N* irá definir o quanto dividida será a frequência do sinal de entrada. Em circuitos digitais, uma maneira fácil de implementar um Divisor de Frequência é por meio de contadores, justamente o que foi utilizado neste trabalho. A figura 29 apresenta o módulo Divisor de Frequência.

3.1.5. Decodificador *FSK*

Para recuperar os dados contidos no sinal *BFSK* é utilizado um *flip-flop* do tipo D em conjunto com o *ADPLL*, onde o primeiro irá atuar como um decodificador binário, tendo na sua saída a informação digital, como exposto na figura 30.

Figura 30 - Módulo Decodificador *FSK*.

Fonte: Autoria própria.

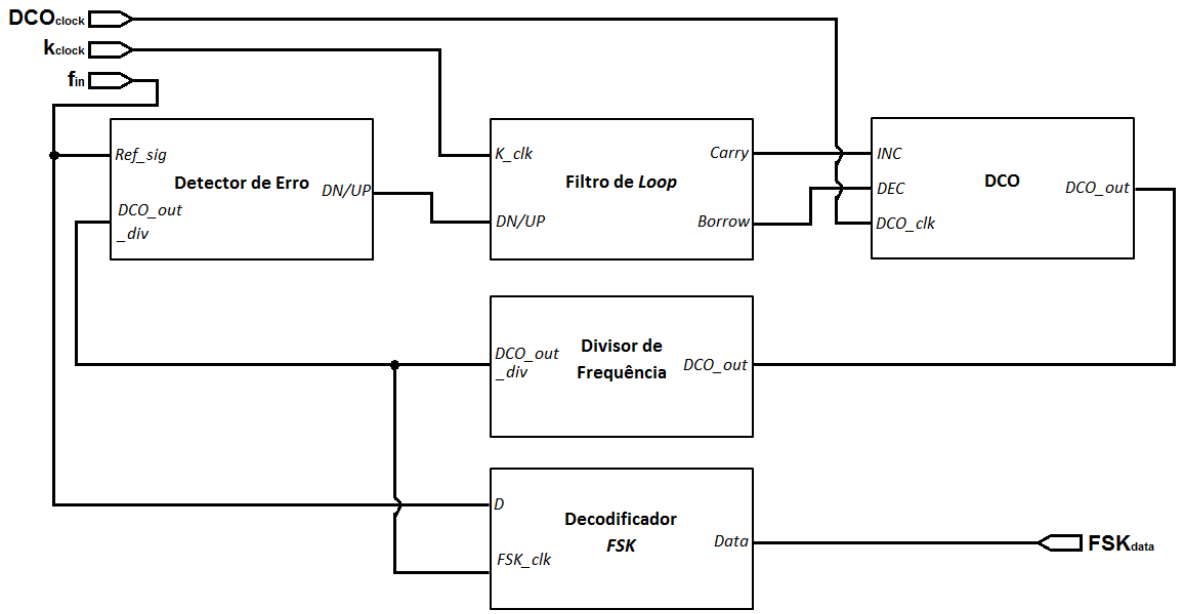
A entrada *D* do decodificador recebe o sinal de referência, sinal modulado *FSK*, e a entrada *FSK_clk* recebe sinal de saída do módulo divisor de frequência, *DCO_out_div*.

A lógica de funcionamento deste bloco consiste basicamente na comparação entre as fases dos sinais de entrada, *D* e *FSK_clk*. Supondo que o sinal modulado *FSK* varie sua frequência entre dois valores, *f1* e *f2*, sendo *fc* uma frequência tal que $f1 < fc < f2$. Estando o sinal *FSK* igual a *f1*, ocorrerá uma diferença de fase negativa entre os sinais de entrada do decodificador, acarretando assim um valor de saída, *Data*, igual a '1'. Analogamente, sendo o sinal *FSK* igual a *f2*, tem-se uma diferença de fase positiva entre os sinais de entrada, logo, gerando uma saída em nível lógico baixo, '0'. Observa-se assim que, a variação da frequência do sinal *FSK* produzirá uma variação entre '1' e '0' na saída do Decodificador *FSK*, que é justamente o dado modulado na portadora.

3.2. DEMODULADOR *BFSK*

Com todos os módulos que compõem o Demodulador *BFSK* implementados logicamente em linguagem de descrição de *hardware*, uniu-se eles a fim de compor um único bloco logico. A forma com que os módulos estão ligados entre si está representada na figura 31.

Figura 31 - Demodulador *BFSK*.



Fonte: Autoria própria.

O Demodulador *BFSK* é alimentado pelo sinal modulado, *Fin*, e pelos pulsos de *clock*, *Kclock* e *DCO*clock. As frequências de oscilação destes pulsos são definidas seguindo alguns parâmetros pré-estabelecidos.

$$Kclock = M * f_c \ (1)$$
$$DCOclock = 2N * f_c \ (2)$$

Onde *M* e *N* são inteiros múltiplos de potência de 2, e *fc* é a frequência central do sinal *FSK* definida pela seguinte equação para um sinal que varia entre as frequências *f1* e *f2*:

$$f_c = \sqrt{f_1 * f_2} \ (3)$$

$$\Delta f_H = |f_2 - f_1| \ (4)$$

Sendo *ΔfH* a amplitude ou variação entre as frequências *f1* e *f2*. Os termos *M* e *N* se relacionam com outro parâmetro, *K*:

$$\Delta f_H = \frac{Mf_c}{2NK} \ (5)$$

Para sistemas demoduladores digitais que utilizam como detector de erro um *ECPD*, a fim de diminuir *ripple*, ou ruído digital, faz-se $M = 2K$, onde é comum fazer $K \geq 8$ para uma melhor performance do *ADPLL*. O valor de *N* é escolhido de forma que ele seja:

$$N \geq Nmin = \frac{3M}{2K} \ (6)$$

Munido dos parâmetros acima estabelecidos, encerrou-se o desenvolvimento do Demodulador *BFSK*.

4. RESULTADOS

Para fins de análise do funcionamento do demodulador *BFSK*, os resultados deste trabalho foram divididos em duas etapas. A primeira etapa consiste na simulação de cada bloco digital separadamente que compõem o demodulador e em seguida a simulação do demodulador como um todo. A segunda etapa dos resultados é a prototipagem do demodulador *BFSK* em *FPGA* e a verificação do seu funcionamento.

4.1. SIMULAÇÕES

Para a simulação do projeto, foi utilizado o *software ModelSim 10.1d* da *Altera*. Inicialmente, cada bloco lógico integrante do demodulador fora testado separadamente para verificar se seu funcionamento estava condizente com o que projetado.

4.1.1. Detector de Erro

Com o objetivo de verificar o funcionamento do Detector de Erro, foram inseridos dois sinais nas entradas do bloco. Estes sinais foram configurados para terem uma diferença de fase entre si, e consequentemente, o módulo deve produzir em sua saída um sinal proporcional a esta diferença detectada entre os sinais de entrada.

Figura 32 - Simulação do Detector de Erro.

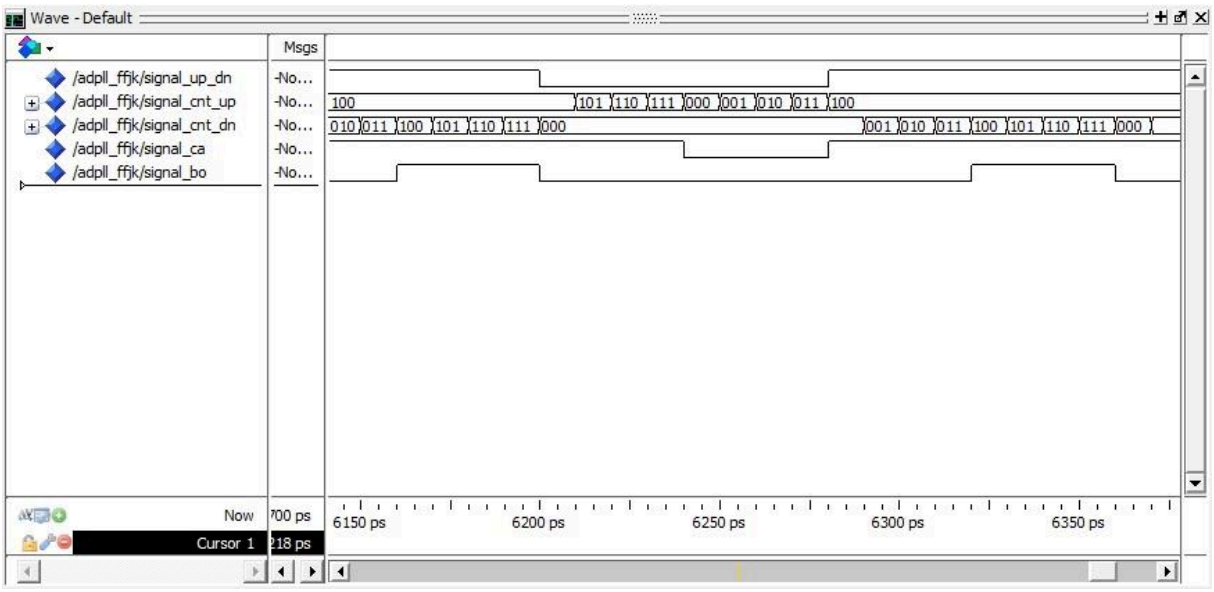
Fonte: Autoria própria.

Na figura 32, verifica-se assim o correto funcionamento do bloco, tendo em vista que este fora programado para responder ao erro entre os sinais de entrada apenas na borda de descida, seja sinal modulado (representado por *v1*) ou do sinal vindo do Divisor de Frequência (exposto na simulação como *v2_div*).

4.1.2. Filtro de *Loop*

Na análise da simulação do Filtro de *Loop*, foi configurado um sinal aleatório para a sua entrada e em seguida verificado a resposta dos dois contadores internos. A figura 33 mostra as formas de onda dos sinais durante a simulação.

Figura 33 - Simulação do Filtro de *Loop*.



Fonte: Autoria própria.

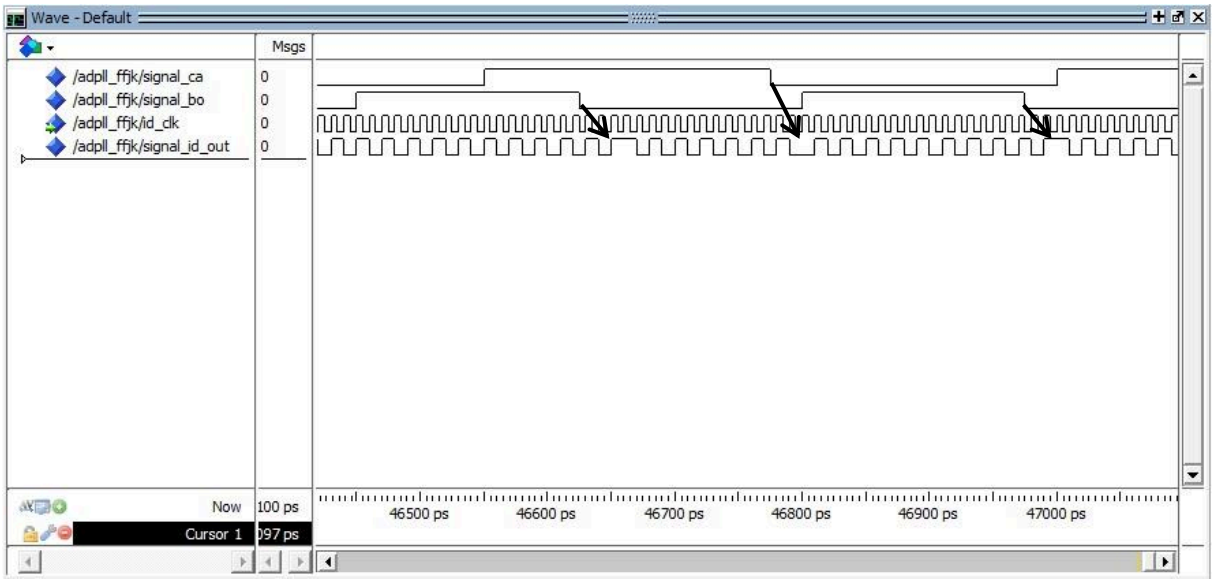
É possível verificar que, enquanto o sinal de entrada (*signal_up_dn*) está em alto, a contagem (representada por *signal_cnt_up*) está ativada, e que, a partir do momento que esta contagem atinge seu valor intermediário, no caso *100* em binário, a saída *Carry* (*ca*) do filtro é ativada. De maneira análoga, enquanto *signal_up_dn* está em nível lógico *0*, a contagem *sinal_cnt_dn* está ativada, gerando um valor alto na saída *Borrow* (*bo*) sempre que a contagem atinge seu valor médio. É importante notar que, ambas as contagens quando atingem seu valor máximo (*111* em binário) são reiniciadas, e consequentemente, as saídas do filtro são postas em nível lógico baixo.

4.1.3. *DCO*

Para a simulação do módulo *DCO* foram gerados sinais arbitrários para as entradas *Carry* e *Borrow*, com isso, pode-se ver o funcionamento do oscilador atuando quase que simultaneamente nos dois modos, adicionando e retirando meio ciclo de *clock* no sinal de saída.

Na figura 34 é apresentada a simulação do *DCO*. Nota-se que, a borda de descida do sinal *Borrow* (representado por *signal_bo*) gera uma adição de meio ciclo de *clock* no sinal de saída do oscilador (*signal_id_out*), e que, a borda de descida do *Carry* (*ca*) resulta em uma retirada de meio ciclo no sinal de saída do *DCO*.

Figura 34 - Simulação do módulo *DCO*.



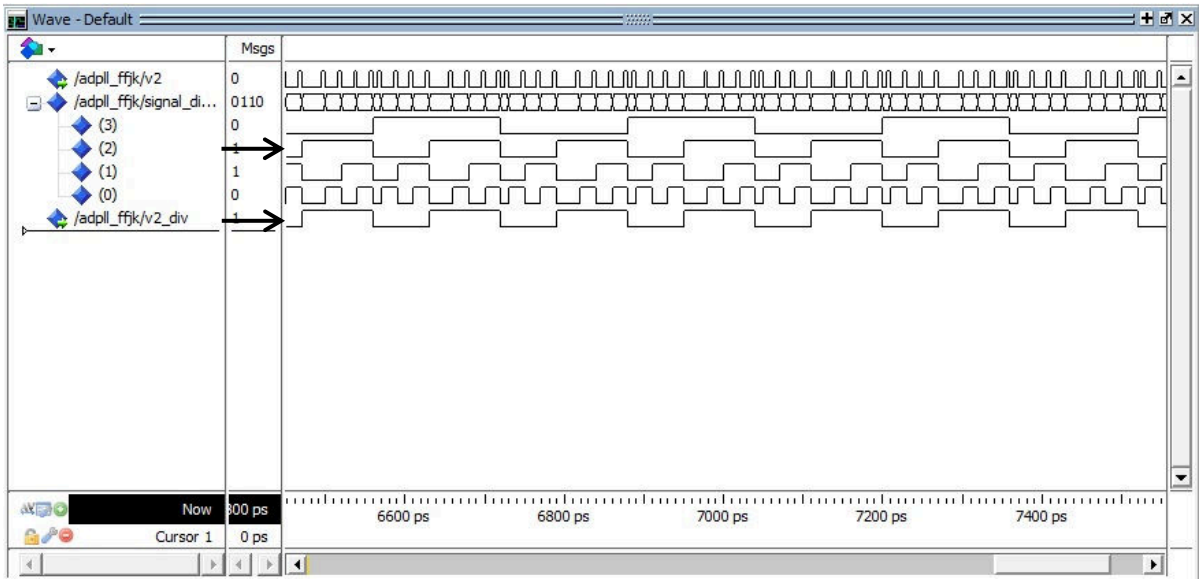
Fonte: Autoria própria.

4.1.4. Divisor de Frequência

O módulo Divisor de Frequência foi simulado colocando-se um sinal oscilante em sua entrada, representada por *v2* na figura 35. A lógica interna implementada para este bloco faz com que a frequência do sinal de entrada seja dividida por múltiplos de potência de dois, $2^1, 2^2, 2^3$ *e* 2^4 por exemplo.

O sinal de saída do bloco, *v2_div*, será uma dos sinais derivados de *v2*, neste caso, com base nos parâmetros *K*, *M* e *N* estabelecidos para este *ADPLL*, o sinal de saída foi o com frequência dividida por 2^3, como é visto na simulação.

Figura 35 - Simulação do módulo Divisor de Frequência.

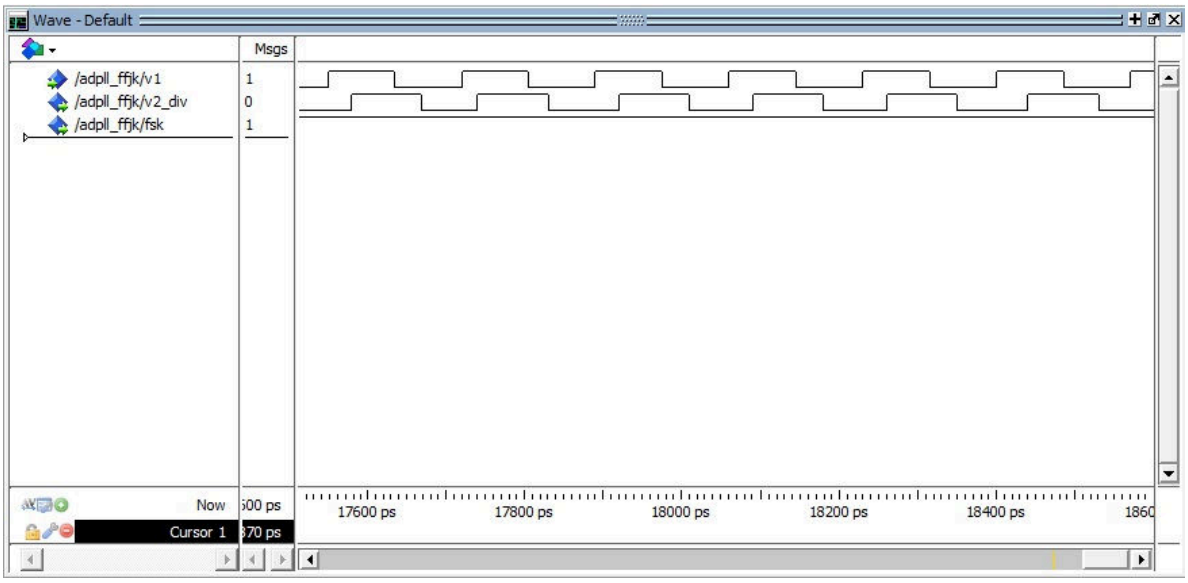


Fonte: Autoria Própria.

4.1.5. Decodificador *FSK*

Este bloco é responsável por extrair os dados da relação entre o sinal modulado e o sinal gerado pelo *ADPLL*. Ele funciona verificando a diferença de fase entre os sinais de entrada e gerando na sua saída o *bit 1* ou *0* de acordo com essa diferença, seja ela positiva ou negativa.

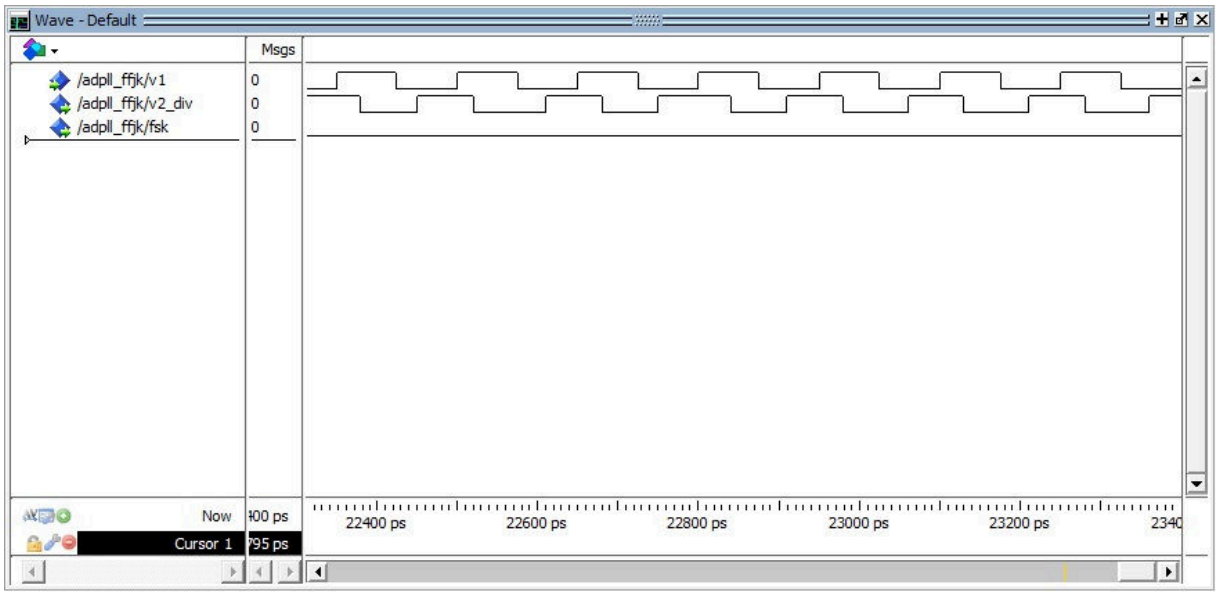
Figura 36 - Simulação do Decodificador *FSK*.



Fonte: Autoria própria.

Para fins de simulação, foi feita apenas a verificação do sinal de saída quando houvesse uma diferença de fase positiva ou negativa nos sinais de entrada. Esta simulação está representada nas figuras 36 e 37, com diferenças de fase positivas e negativas respectivamente.

Figura 37 - Simulação do Decodificador *FSK*.



Fonte: Autoria própria.

4.1.6. Demodulador *BFSK*

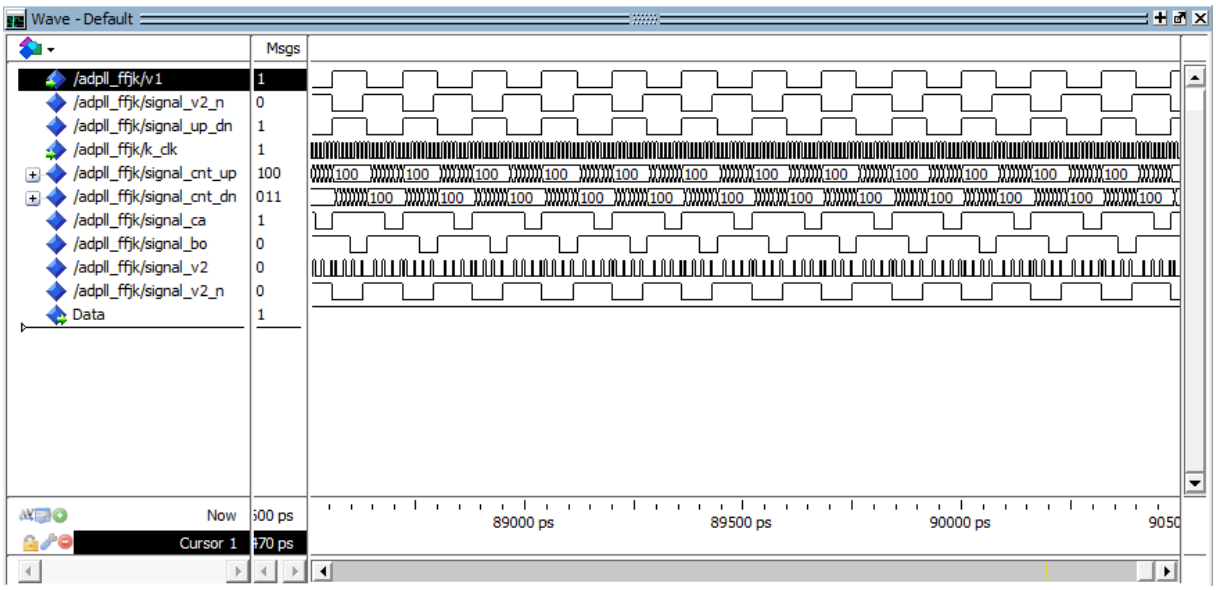
Para a simulação do Demodulador *BFSK* foram escolhidos os seguintes parâmetros: $K = 8, N = 8, M = 16$, um sinal modulado com frequência central $f_c = 6{,}25\ GHz$ e com um *range* $\Delta f_h = 0{,}5\ GHz$, desta forma, o sinal *BFSK* varia entre $f_1 = 6{,}00\ GHz$ e $f_2 = 6{,}50\ GHz$. A análise do funcionamento do *ADPLL* atuando como um Demodulador *BFSK* foi feita em duas partes.

A primeira análise foi feita quando o sinal modulado estava com sua frequência igual à frequência central f_c, como mostrado na figura 38. Nota-se que o *ADPLL* trava na frequência central de funcionamento, e que em decorrência disto, todos os seus sinais internos permanecem constantes ou se repetem periodicamente.

A segunda análise consistia em verificar o funcionamento do *ADPLL* como bloco demodulador enquanto um sinal *BFSK* era injetado em *v1*. A figura 39 mostra a resposta do Demodulador *BFSK* quando o sinal *BFSK* estava com frequência igual a f_2, representando

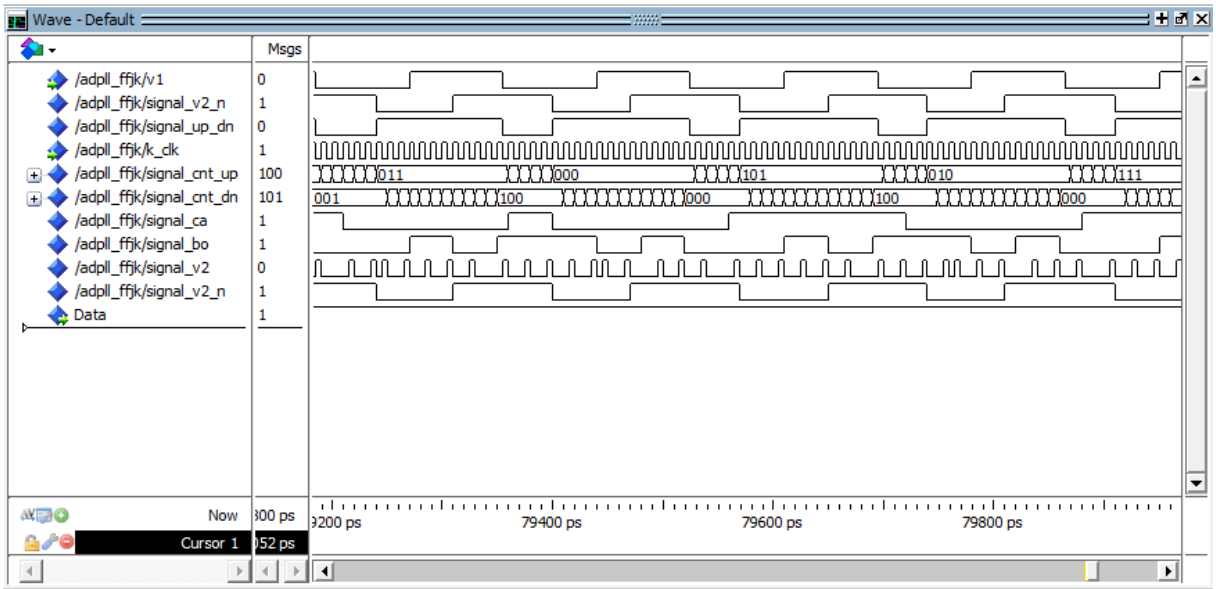
assim nível lógico alto. A figura 40 mostra a resposta do demodulador quando o sinal modulado estava com frequência igual a f_1, representando nível lógico baixo. E finalmente, a figura 41 representa a simulação do Demodulador *BFSK* exposto a um sinal *BSFK* com a frequência variando entre f_1 e f_2.

Figura 38 - Simulação do Demodulador *BFSK* atuando na frequência central.



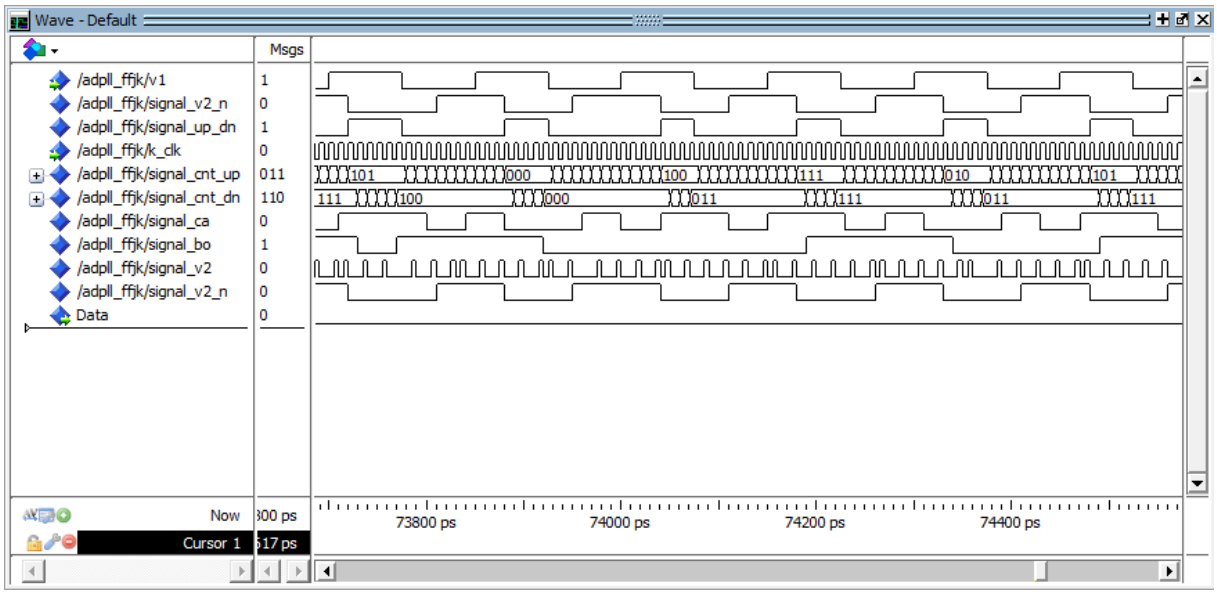
Fonte: Autoria própria.

Figura 39 - Simulação do Demodulador *BFSK* atuando na frequência de nível alto.



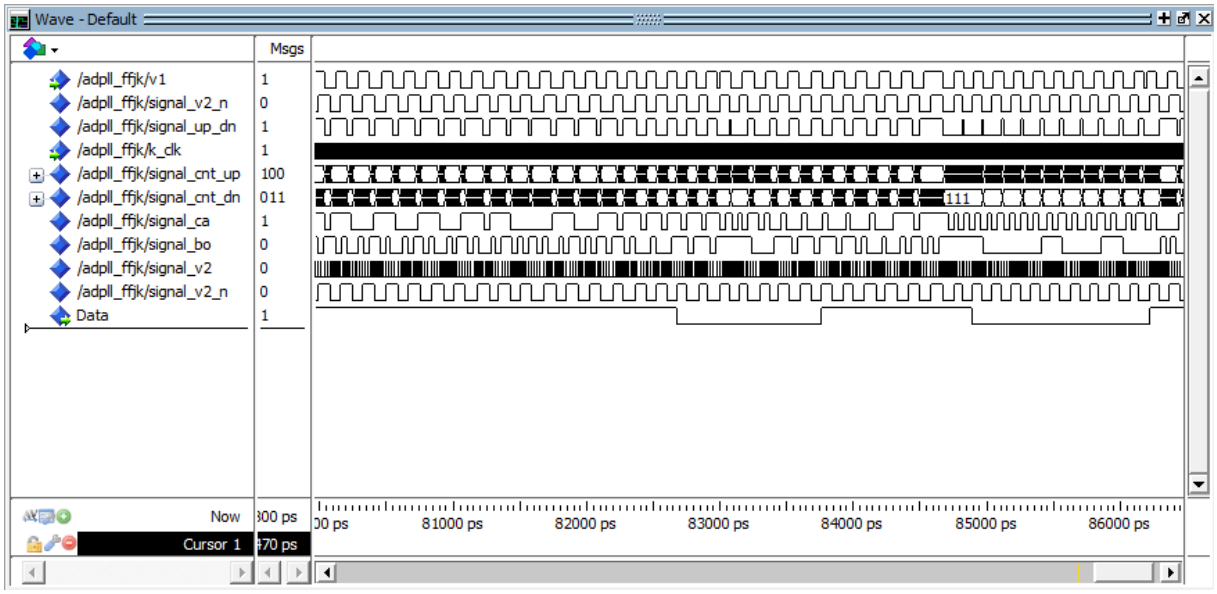
Fonte: Autoria própria.

Figura 40 - Simulação do Demodulador *BFSK* atuando na frequência de nível baixo.



Fonte: Autoria própria.

Figura 41 - Simulação do Demodulador *BFSK* atuando em ambos os níveis de frequência.



Fonte: Autoria própria.

Percebe-se assim, que, ao receber um sinal modulado com frequência f_1, o demodulador responde gerando um *bit 0* em sua saída, e, ao receber um sinal modulado com frequência f_2, o demodulador gera em sua saída um *bit* igual a *1*. Os *bits 1* e *0* são justamente as informações moduladas no sinal *BFSK*, desta forma, comprova-se o correto funcionamento do Demodulador *BFSK* projetado e simulado.

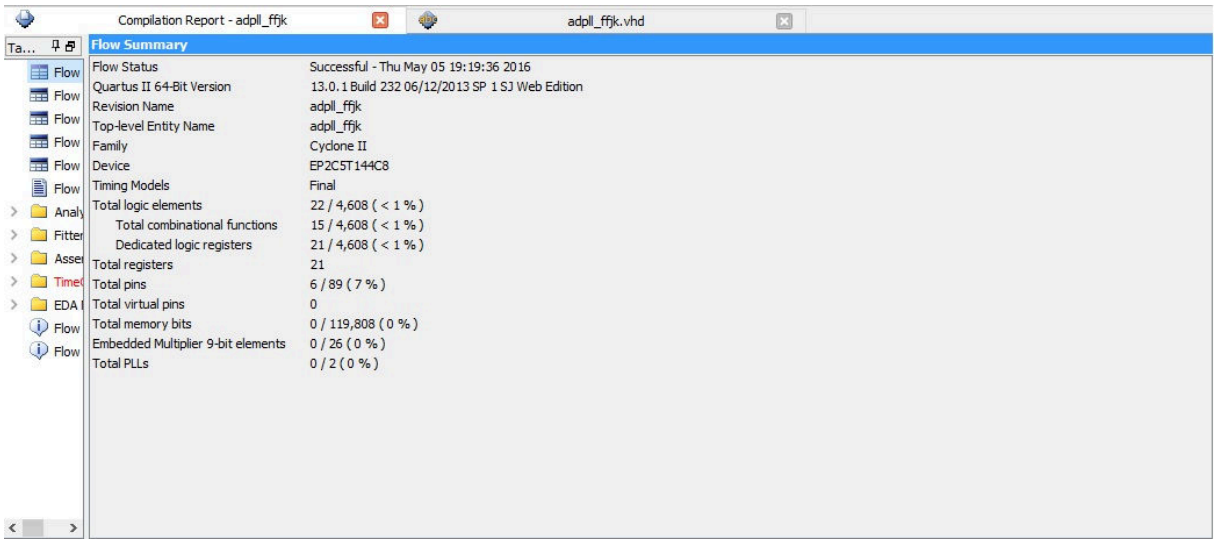
4.2. PROTOTIPAGEM

Após a simulação de cada módulo e do demodulador como um todo, o código escrito em linguagem de descrição de *hardware* foi compilado utilizando o *software* de desenvolvimento *Quartus II 13.0* da *ALTERA* e, sem seguida, programado no *FPGA*.

Foi utilizado um *kit* de desenvolvimento com um *FPGA* de modelo *EP2C5* da família *Cyclone II*, contendo 4680 células programáveis, em conjunto com um oscilador híbrido de *20 MHz*. Este modelo contém ainda 144 pinos, sendo 87 desses disponíveis para conexões externas.

A figura 42 mostra os resultados da compilação do código para o *FPGA* utilizado. Nota-se que foram utilizados apenas 22 elementos lógicos dos 4680 disponíveis.

Figura 42 - Relatório de compilação do código.



Fonte: Autoria própria.

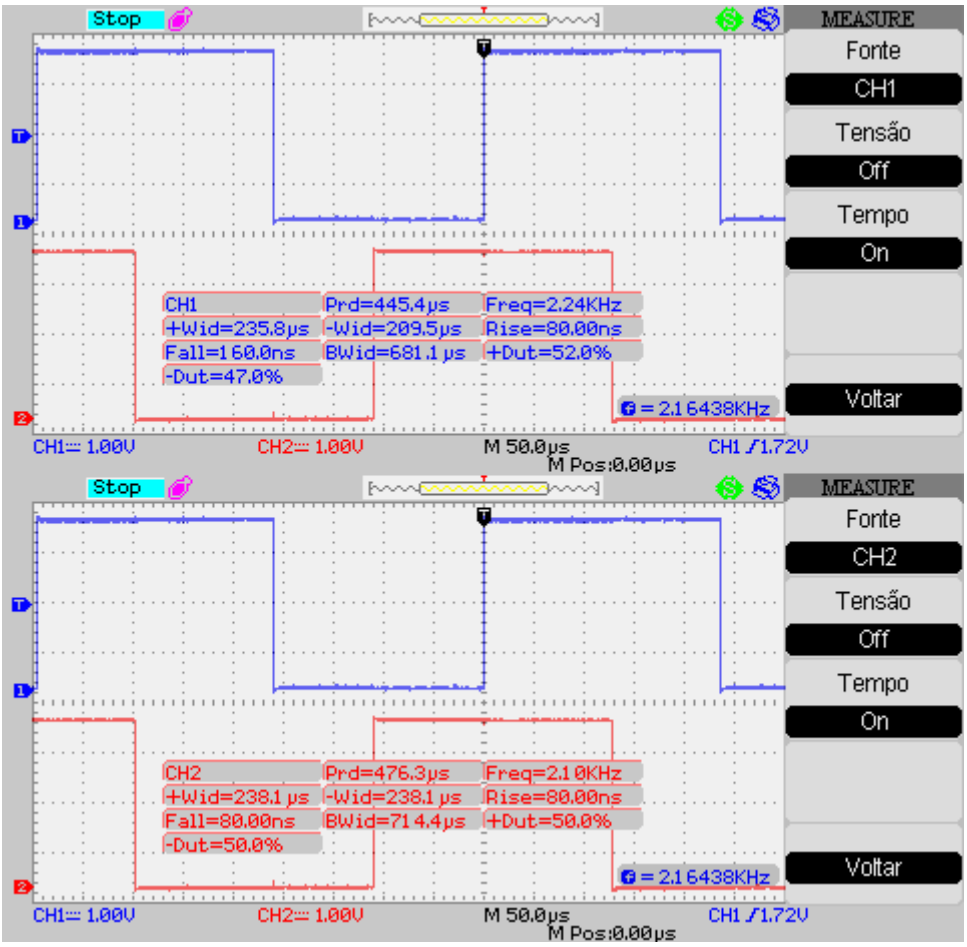
Após a compilação e programação do código no *FPGA*, deu-se inicio a fase de testes. Estes foram realizados no laboratório de Engenharia Elétrica da UFERSA Campus Caraúbas, onde foi utilizado um osciloscópio para análise das formas de onda dos sinais gerados pelo Demodulador *BFSK*.

Para a verificação do correto funcionamento do código implementado no *FPGA*, foi sintetizado outro código em linguagem de descrição de *hardware* para produzir o sinal que faria o papel do sinal modulado, e que produziria os *clocks* necessários para o funcionamento do *ADPLL*. Este código foi incorporado ao do demodulador e compilado em conjunto.

A partir do oscilador disponível no *kit* de desenvolvimento, foram sintetizados três sinais. Os dois primeiros para simularem o papel de um sinal modulado na frequência, com a frequência de *2,1 KHz* representando o *bit 0* e a frequência de *2,7 KHz* para o *bit 1*. O terceiro sinal produzido, de frequência igual a *38,4 KHz*, serviu de alimentação para o *ADPLL*, mais precisamente para o Filtro de *Loop* e *DCO*. Os valores destas frequências foram calculados a partir dos parâmetros pré-estabelecidos (*K*, *M* e *N*) e em relação à frequência central do sinal modulado, aproximadamente igual a *2,4 KHz*.

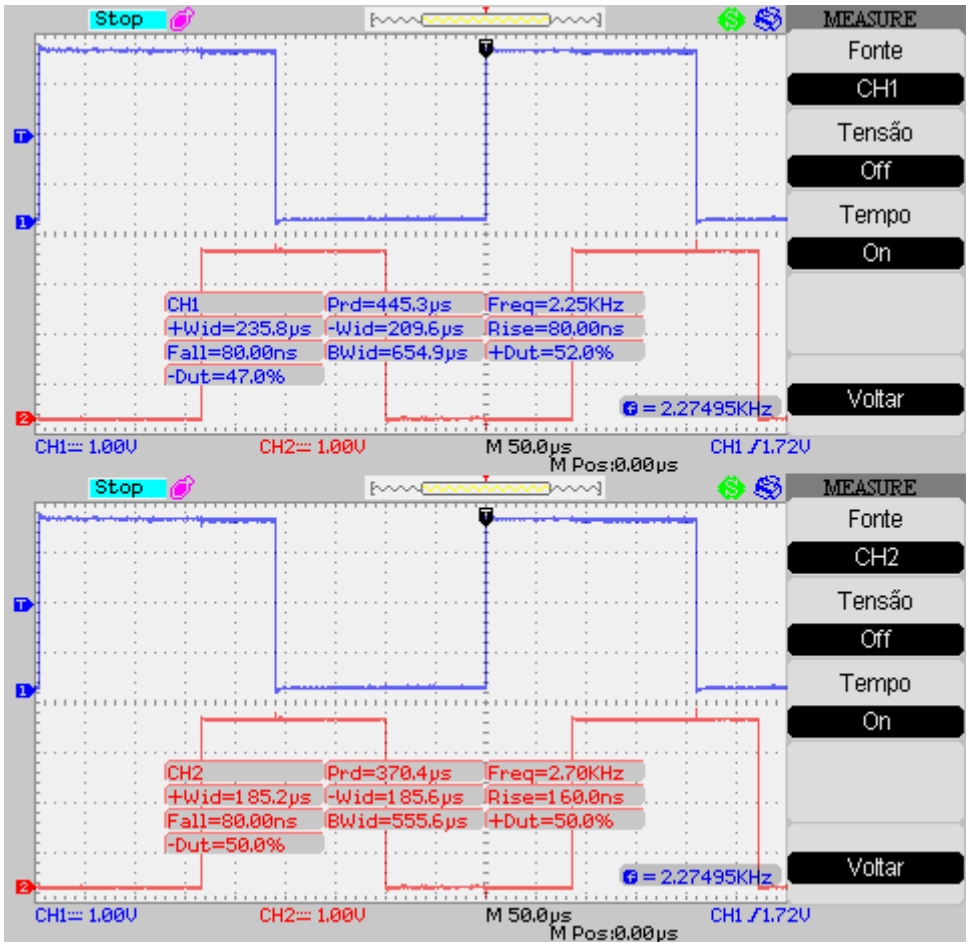
Primeiramente, buscou-se verificar se o *ADPLL* estava atuando na faixa central de frequência projetada, aproximadamente igual a *2,4 KHz*, e analisar se o código gerador de frequência estava funcionando ou não. Nas figuras 43 e 44 estão representadas as formas de onda dos sinais. Em azul está o sinal gerado pelo *ADPLL* e em vermelho o sinal gerado pelo código de teste.

Figura 43 - Teste 1, verificação do sinal do *ADPLL* e da frequência de nível lógico baixo.



Fonte: Autoria própria.

Figura 44 - Teste 2, verificação do sinal do ADPLL e da frequência de nível lógico alto.



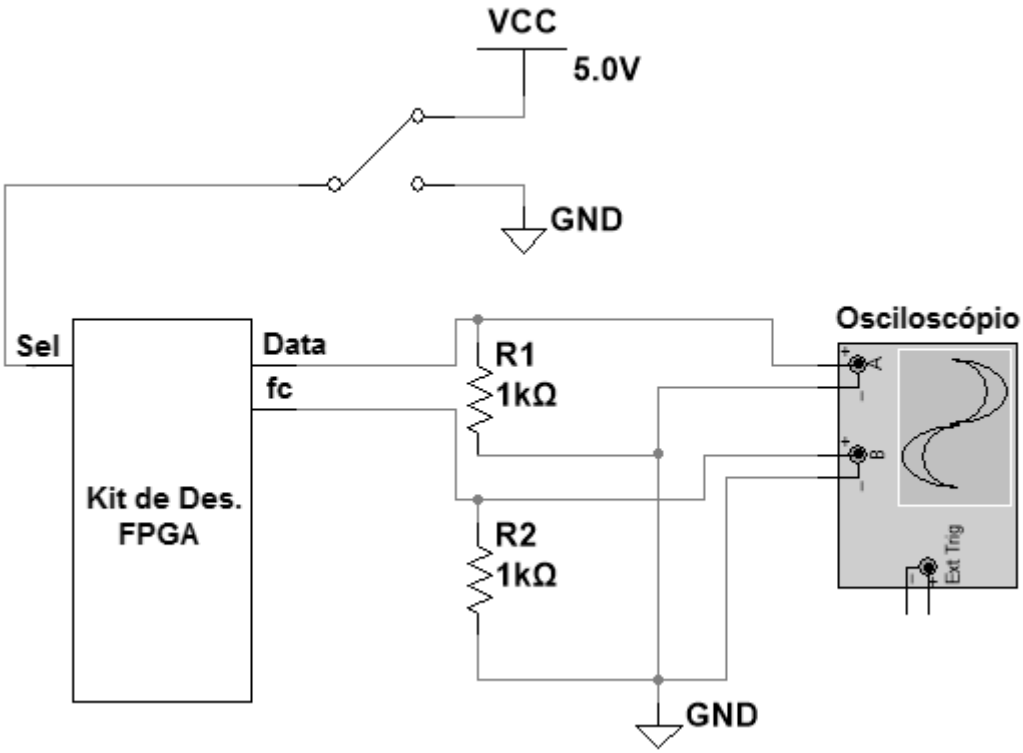
Fonte: Autoria própria.

Na figura 43, apresentado em azul, verifica-se a frequência do sinal gerado pelo *ADPLL*, f_c, igual *2,24 KHz*, diferenciando-se um pouco de *2,4 KHz*, que fora o f_c projetado, entretanto, durante os testes esta medição oscilou bastante, chegando a ultrapassar a faixa dos *2,4 KHz*, pode-se supor então algum fator que interferiu nesta medição, como ruído do circuito interno e entre outros. Nota-se, também, a frequência de *2,1 KHz*, exposta em vermelho, correspondente ao nível baixo do sinal modulado. Semelhante, na figura 44 estão presentes os mesmo resultados, diferenciando-se apenas no sinal, exposto em vermelho, do nível alto de frequência, correspondente a *2,7 KHz*.

Dando sequência aos testes, alterou-se a frequência de entrada do *ADPLL* simulando a variação de frequência de um sinal modulado na frequência. A figura 45 mostra o esquema montado para realização do teste. A figura 46 consta os resultados para frequência modulada igual a *2,1 KHz*. Comprova-se o correto funcionamento do demodulador nestas circunstâncias

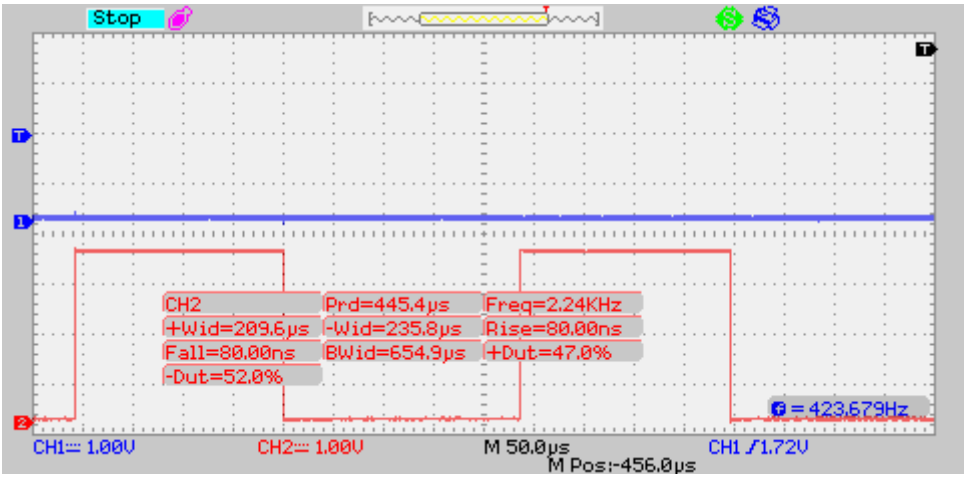
quando o resultado da saída deste é igual ao nível lógico baixo, em azul, correspondente ao sinal injetado de frequência menor.

Figura 45 - Esquema montado para teste.



Fonte: Autoria Própria.

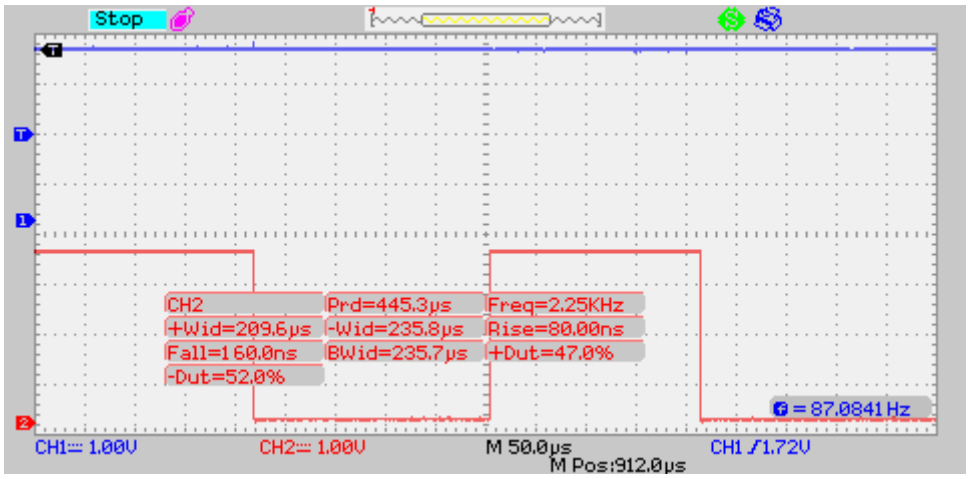
Figura 46 - Teste 3, demodulador recebendo a frequência correspondente ao nível lógico baixo.



Fonte: Autoria própria.

Finalizando as análises e testes do demodulador, aplicando-se um sinal de frequência de *2,7 KHz*, representando o nível lógico alto, verificou-se o funcionamento do demodulador. Na figura 47 estão os resultados para este caso, percebe-se, em azul, o nível lógico alto do sinal já demodulado.

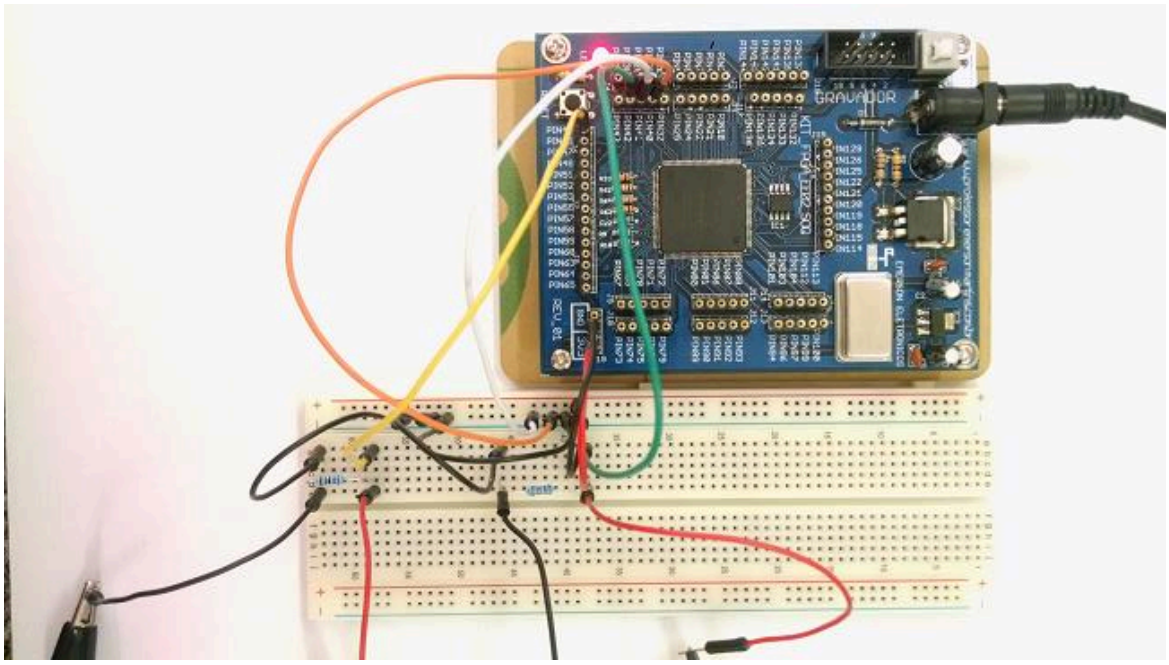
Figura 47 - Teste 4, demodulador recebendo a frequência correspondente ao nível lógico alto.



Fonte: Autoria própria.

A figura 48 mostra o circuito montado para realização dos testes com o *FPGA*.

Figura 48 - Esquema com *FPGA* montado para os testes.



Fonte: Autoria própria.

5. CONCLUSÃO E TRABALHOS FUTUROS

A partir dos estudos e análises realizadas por meio de levantamentos bibliográficos, simulações e experimentos, foi possível compreender as noções básicas a cerca dos sistemas de modulação mais usuais, enfatizando a modulação de frequência. Com o desenvolvimento do código em linguagem de descrição de *hardware* e posteriormente a implementação em *FPGA*, foi possível vivenciar a metodologia de desenvolvimento de um sistema digital, desde seu projeto inicial, passando pela análise de simulações, até seu teste experimental.

Os resultados obtidos tanto nas simulações como nos testes realizados com o *FPGA* comprovaram o correto funcionamento do demodulador de acordo com os parâmetros estabelecidos no projeto, sendo possível a visualização da informação na saída do demodulador, antes modulada na portadora.

Este trabalho se apresenta como um estágio inicial, servindo de base para projetos futuros mais complexos. Vislumbram-se as seguintes melhorias para este trabalho:

- Otimizar o código para possibilitar a atuação do demodulador em faixas maiores de frequência.
- Desenvolver um Modulador *BFSK* e possível implementação de um *Modem* embarcado em *FPGA*.
- Desenvolver um rádio definido por *software*, sintetizado a partir de linguagem de descrição de *hardware* e implementado em *FPGA*.

6. REFERENCIAS

AGILENT TECHNOLOGIES (United States). **Digital Modulation in Communications Systems** — An Introduction: Application Note 1298. 2001. Disponível em: <http://cp.literature.agilent.com/litweb/pdf/5965-7160E.pdf>. Acesso em: 01 abr. 2016.

AL-ARAJI, Saleh R.; HUSSAIN, Zahir M.; AL-QUTAYRI, Mahmoud A.. **DIGITAL PHASE LOCK LOOPS**: Architectures and Applications. [s.i.]: Springer Us, 2006. 192 p.

BEST, Roland E.. **Phase-Locked Loops:** Design, Simulation, and Applications.. 6. ed. New York: Mcgraw-hill Education, 2007.

CARVALHO, Rogério Muniz. **Comunicações Analógicas e Digitais**. Rio de Janeiro: Ltc, 2009.

CAVALCANTI, José H. F.; FERREIRA, José R. S.; ALSINA, Pablo J. **Conversor Analógico Digital Usando Dipolos De Hopfield**. In: IV SBAI – Simpósio Brasileiro de Automação Inteligente, São Paulo - SP, Setembro 1999.

COSTA, Cesar da. **Projetos de Circuitos Digitais Com Fpga.** 3. ed. São Paulo: Editora Érica, 2014.

ELSEROUGI, Kristen. **PHASE LOCKED LOOP DESIGN**. 2006. 49 f. TCC (Graduação) - Curso de Bachelor Of Science In Electrical Engineering, School Of Engineering Santa Clara University, Santa Clara, Ca, 2006. Disponível em: <https://www.academia.edu/4940355/PHASE_LOCKED_LOOP_DESIGN_SENIOR_DESIGN_PROJECT_REPORT>. Acesso em: 04 abr. 2016.

FONSECA, Eduardo A. D; LIMA, Luiz A. P. O **Papel Dos Conversores Sigma-Delta No Front End Dos Sistemas De Comunicação Digital**. In: <www.revdigonline.com> acesso: 10, Outubro 2014. Vol. 5, agosto 2005.

GOMES, Alcides Tadeu. **Telecomunicações: Transmissão e Análise**. 21. ed. São Paulo: Érica, 2013.

KILTS, Steve. **Advanced FPGA Design:** Architecture, Implementation, and Optimization. [s.i.]: John Wiley & Sons, 2007.

LATA, Kusum; KUMAR, Manoj. ALL Digital Phase**-Locked Loop (ADPLL)**: A Survey. International Journal Of Future Computer And Communication. [s.i.], p. 551-555. dez. 2013. Disponível em: <http://www.ijfcc.org/papers/225-E353.pdf>. Acesso em: 07 abr. 2016.

MILAGRE JÚNIOR, Joaquim. **Análise do comportamento de uma PLL**. 2001. 51 f. TCC (Graduação) - Curso de Engenharia Elétrica, Universidade do Porto, Porto, 2001. Disponível em: <https://web.fe.up.pt/~ee97148/etele/pll.pdf>. Acesso em: 04 abr. 20016.

PATEL, Mahendra. I**mplementation of FSK Modulation and Demodulation using CD74HC4046A**. Houston: Texas Instruments Incorporated, 2013. Disponível em: <http://www.ti.com/logic/docs/litabsmultiplefilelist.tsp?sectionId=452&tabId=1989&literatureNumber=slaa618&docCategoryId=1&familyId=1>. Acesso em: 25 jan. 2016.

SOARES NETO, Vicente. **Telecomunicações Sistemas de Modulação**: Uma visão sistêmica. 3. ed. São Paulo: Érica, 2012.

TONGUZ, O.k.; WAGNER, R.e.. **Equivalence between preamplified direct detection and heterodyne receivers**. Ieee Photonics Technology Letters, [s.l.], v. 3, n. 9, p.835-837, set. 1991. Institute of Electrical & Electronics Engineers (IEEE). http://dx.doi.org/10.1109/68.84510.

YOUNG, Paul H.. **Técnicas de Comunicação Eletrônica**. 5. ed. São Paulo: Pearson Prentice Hall, 2006.

ANEXO A – **CÓDIGO EM *VHDL* DO DEMODULADOR *BFSK*.**

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;
use ieee.std_logic_misc.all;

entity adpll_ffjk is
     generic (K: integer := 8;
              N: integer := 4;
              i: integer := 1);
     port (
     v1:       in   std_logic;
     k_clk:    in   std_logic;
     id_clk:   in   std_logic;
     fsk:      out  std_logic;
     v2:       out  std_logic;
     v2_div:   out  std_logic);
end adpll_ffjk;

architecture arch of adpll_ffjk is
signal signal_up_dn:       std_logic;
signal signal_ca:          std_logic;
signal signal_bo:          std_logic;
signal signal_id_out:      std_logic := '0';
signal signal_v2_n:        std_logic;
signal signal_v2:          std_logic;
signal signal_q:           std_logic;
signal  signal_cnt_up:     std_logic_vector  (K-6  downto  0)  :=
(others => '0');
signal signal_cnt_dn:      std_logic_vector  (K-6  downto  0)  :=
(others => '0');
signal c1, c2, c3, c4:     std_logic;
```

```
signal b1, b2, b3, b4:     std_logic;
signal signal_div_n:       std_logic_vector(N-1  downto  0)  :=
(others => '0');
signal pd1, pd2, rpd:       std_logic;
signal f_in, f_out:        std_logic;
signal signal_fsk, d:      std_logic;
signal clk_fsk:           std_logic;

begin

-- ecpd_ff_jk -- Detector de Erro
f_in <= v1;
f_out <= signal_v2_n;
rpd <= pd2;
     process(rpd,f_out)
     begin
          if rpd = '1' then
               pd1 <= '0';
          elsif f_out'event and f_out = '0' then
               pd1 <= '1';
          end if;
     end process;
     process(rpd,f_in)
     begin
          if rpd = '1' then
               pd2 <= '0';
          elsif f_in'event and f_in = '0' then
               pd2 <= '1';
          end if;
     end process;
signal_q <= pd1;

-- k_counter - Loop Filter
signal_up_dn <= signal_q;
```

```
     process (k_clk)
     begin
          if k_clk'event and k_clk = '1' then
               if signal_up_dn = '0' then
                    signal_cnt_up <= signal_cnt_up + 1;
                    signal_cnt_dn <= signal_cnt_dn;
               elsif signal_up_dn = '1' then
                    signal_cnt_dn <= signal_cnt_dn + 1;
                    signal_cnt_up <= signal_cnt_up;
               end if;
          end if;
     end process;
signal_ca <= signal_cnt_up(K-6);
signal_bo <= signal_cnt_dn(K-6);

-- id_counter -- DCO
     process(id_clk)
     begin
          if id_clk'event and id_clk = '1' then
               c1 <= signal_ca;
               c2 <= c1;
               c3 <= c2;
               c4 <= ( (not c1) and c2 and signal_id_out ) or (
(not c2) and c3 and signal_id_out );
               b1 <= signal_bo;
               b2 <= b1;
               b3 <= b2;
               b4 <= ( (not b1) and b2 and (not signal_id_out)
) or ( (not b2) and b3 and (not signal_id_out) );
               if c4 = '1' and b4 = '1' then
                    signal_id_out <= signal_id_out;
               elsif c4 = '0' and b4 = '1' then
                    signal_id_out <= '1';
               elsif c4 = '1' and b4 = '0' then
```

```
                    signal_id_out <= '0';
               elsif c4 = '0' and b4 = '0' then
                    signal_id_out <= (not signal_id_out);
               end if;
          end if;
     end process;
     signal_v2 <= (not signal_id_out) and (not id_clk);

-- div_n -- Divisor de Frequência
     process(signal_v2)
     begin
          if signal_v2'event and signal_v2 = '0' then
               signal_div_n <= signal_div_n + 1;
          end if;
     end process;
signal_v2_n <= signal_div_n(N-2);

-- fsk_decorder(ff_d)-- Decodificador
clk_fsk <= signal_v2_n;
d <= v1;
     process(clk_fsk)
     begin
          if (clk_fsk'event and clk_fsk = '1') then
               signal_fsk <= d;
          else
               signal_fsk <= signal_fsk;
          end if;
     end process;
fsk <= signal_fsk;

v2 <= signal_v2;
v2_div <= signal_v2_n;
end arch;
```